Ingeniería de Telecomunicación.

PROYECTO FIN DE CARRERA

Desarrollo de un asistente multimodal para proporcionar información sobre la televisión y el cine en dispositivos Android

Autor: Irene Marquet Gómez

Tutor/Director: David Griol Barres

Leganés, Junio de 2014

Título: Desarrollo de un asistente multimodal para proporcionar información sobre la televisión y el cine en dispositivos Android
Autor: Irene Marquet Gómez
Director: David Griol Barres

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día ___ de _____ de 20___ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

En primer lugar quiero mostrar mi agradecimiento a mi tutor, David Griol, por su ayuda y apoyo durante el desarrollo de este proyecto, y sobre todo, por la confianza que ha depositado en mí. Gracias a sus palabras llenas de optimismo he disfrutado al máximo con la elaboración del proyecto y he sentido que estaba desarrollando algo valioso.

En segundo lugar, quiero expresar mi gratitud a toda mi familia. En especial a mi madre y hermana por ser como son, por estar conmigo en mis mejores momentos y apoyarme en los peores. Sin ellas la realización del proyecto hubiese sido imposible. Agradecimientos también a mi padre y a mi abuelito, por todas las oportunidades que me han brindado desde que soy pequeña. También gracias al resto de mi familia, por parte de madre y padre, que son todos maravillosos. Finalmente, no puedo olvidarme de Pep, por haber estado en todo momento a mi lado, siendo mi sombra y logrando que nunca me sintiese sola. Gracias por conformarte con mi simple compañía y mostrar la fidelidad más grande que puede existir sin pedir nada a cambio.

Me gustaría dar un agradecimiento muy especial a Ion. Gracias por haber tenido tanta paciencia y por haberme animado constantemente. Sin tu apoyo no hubiese conseguido terminar este trabajo. Quiero agradecer de la misma manera el apoyo ofrecido por las personas de tu entorno, especialmente a Mayte, Juan e Irenka.

Gracias a todos mis compañeros de la Universidad, particularmente a los que me han acompañado desde el principio hasta el final. Cada uno de vosotros habéis aportado en mi vida algo maravilloso, sois personas muy especiales y sé que seréis amigos para siempre.

Mis más sinceros agradecimientos también a toda la gente de Deimos ya que os habéis convertido en poco tiempo en gente muy especial en mi vida. He de agradeceros a vosotros el título de mi aplicación y el haberme aportado la inspiración que necesitaba para terminar este proyecto. También sé que muchos de vosotros seréis amigos para siempre.

No puedo olvidarme de Marta y Cris, que aunque estén lejos me han demostrado durante muchísimos años que son amigas increíbles y me han dado su apoyo incondicional en todo. Muchísimas gracias chicas, sois las mejores.

Finalmente, quiero dar las gracias a todas las amistades de mi hermana, que estos últimos años han demostrado ser las mías también. En especial a mis bebes, ya que habéis logrado hacerme sentir como en una familia.

Gracias a todos.

Resumen

En el presente Proyecto Final de Carrera se ha desarrollado una aplicación para dispositivos móviles Android consistente en un asistente virtual multimodal cuya función es la de proporcionar, a través de un sistema de diálogo establecido con el usuario o través de los interfaces tradicionales como el teclado o la pantalla, información televisiva y de cine atendiendo a los gustos registrados por el usuario.

La aplicación para dispositivos móviles Android permite al usuario acceder a distintos servicios a través de la modalidad deseada: consulta de la programación emitida por televisión de acuerdo a los gustos registrados, recomendación de películas de acuerdo a los gustos registrados, búsqueda de información de una película (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios) y búsqueda de la cartelera de una sala de cine concreta.

Gracias a la aplicación desarrollada, se facilita el acceso a los servicios a personas con problemas de visión o discapacidades motoras, o en entornos en los que no es aconsejable o resulta imposible el uso de interfaces tradicionales (por ejemplo, en conducción).

El presente proyecto también incluye un estudio completo de los sistemas de diálogo con el objetivo principal de incorporar a la aplicación para dispositivos móviles Android las características típicas de los sistemas de diálogo ideales y de este modo aprovechar toda la potencia que ofrecen dichos sistemas.

Además, se realiza un estudio exhaustivo de la plataforma Android y del entorno de desarrollo de sus aplicaciones, centrándose en las posibilidades que ofrece dicha plataforma para el desarrollo de aplicaciones que permitan la interacción oral con el usuario, es decir, que integren el reconocimiento automático del habla para la entrada, y la síntesis de texto a voz para la salida. Este estudio de posibilidades ofrecidas por Android para la interacción oral viene acompañado de ejemplos de asistentes virtuales y otras aplicaciones de interés para dispositivos móviles Android, existentes en la actualidad y que son similares a la aplicación que se ha desarrollado para este proyecto.

La aplicación utiliza además otras tecnologías adicionales cuyo estudio se incluye también en este proyecto, como son el desarrollo de una arquitectura cliente-servidor de modo que la aplicación Android pueda acceder a una base de datos MySQL de un servidor web remoto, la utilización de una base de datos SQLite en Android, y como extraer contenido HTML y descargar imágenes de páginas web.

Palabras claves: Sistemas de diálogo, interacción oral, interacción multimodal, asistente televisivo, Android, MySQL, PHP, SQlite, *web scraping*.

Abstract

This Bachelor project aims to develop a multimodal virtual assistant for Android mobile devices that provides access, combining the visual modality (keyboard and screen) and the voice modality (speech recognition for input and speech synthesis for output), to cinema and television information according to the user registered preferences.

The developed virtual assistant for Android mobile devices allows the user to choose the mode of interaction with the device and to access the following services: television program recommendations according to the user registered preferences, movie recommendations according to the user registered preferences, search for information on any movie (synopsis, original title, country, running time, director, cast, genre, year, rating and awards) and search for an updated list of the cinemas and movies in Spain.

Thanks to the developed application, these services can be accessed by people with vision problems or motor disabilities and in environments where it is difficult or impossible to use traditional interfaces (E.g. driving).

This project also includes a full study of spoken dialog systems in order to provide to the application for Android mobile devices the distinctive features of a perfect spoken dialog system and therefore take advantage of the capabilities offered by these systems.

Moreover, this project includes a thorough study on the Android platform and the development environment for Android applications, focusing in the alternatives offered by this platform to develop applications that allow oral interaction (i.e. speech recognition for the user input and speech synthesis for the user output). This study also includes examples of other virtual assistants and applications of interest, currently available and similar to the application that has been developed in this project.

The developed application also requires additional technologies so this project also includes a study in how to develop client-server architectures so that the Android application can access a MySQL database in a remote server, how to manage a SQLite database in Android mobile devices, and how to parse html content and download images from web pages.

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

INTRODUCCIÓN

Este primer capítulo comienza con una introducción del presente Proyecto Final de Carrera, que incluye una breve descripción de los sistemas de diálogo y de sus limitaciones y antecedentes, con el objetivo de justificar la motivación de dicho proyecto de desarrollar una aplicación multimodal para dispositivos móviles Android. Posteriormente, se describen los objetivos del proyecto y se establecen las fases de desarrollo y la lista de medios empleados para su desarrollo. Por último, se resume el contenido de cada capítulo que forma parte de la estructura del presente documento.

1.1. Introducción

Los sistemas de diálogo hablado (*spoken dialogue systems*) son una tecnología concebida para facilitar la interacción natural mediante el habla entre un usuario y un sistema informático, proporcionando la ventaja de no requerir el uso de interfaces tradicionales como el teclado, el ratón o la pantalla y de recurrir, en cambio, al medio de comunicación propio de los seres humanos. Por ello, estos sistemas se utilizan en aquellos casos en los que la lengua oral permite llevar a cabo una determinada tarea de forma más flexible y más eficaz que la escrita.

Para que un sistema de diálogo sea ideal es necesario que reconozca sin dificultades el habla espontánea; comprenda enunciados sin restricciones de contenido; proporcione respuestas con sentido, gramaticalmente bien formadas y pragmáticamente adecuadas; responda con voz completamente natural; y que permita una interacción multimodal a través del habla y de la imagen.

No obstante, a pesar de que la investigación ha avanzado notablemente en las últimas décadas, construir sistemas de diálogo ideales constituye un reto ya que están sujetos a las limitaciones del reconocimiento automático del habla; su comprensión y respuesta están restringidas a dominios específicos; se encuentran condicionados por la naturalidad del habla sintetizada; aun sigue existiendo la necesidad de que el interlocutor humano guíe al sistema con el fin de confirmar que la interpretación de sus preguntas es la correcta; y debido a problemas del diálogo espontáneo (Llisterri, 2006).

Con el objetivo de evitar las limitaciones de la interacción basada exclusivamente en el habla surgen los sistemas de diálogo multimodales. En una interacción multimodal el usuario no está restringido a utilizar el habla como único canal de comunicación, sino que puede utilizar varios dispositivos de entrada, como por ejemplo un teclado, un ratón, un micrófono, una cámara, una pantalla sensible al tacto, etc. Asimismo, un sistema multimodal puede utilizar diversos canales de salida para proporcionar información al usuario como por ejemplo, voz, texto, gráficos o imágenes, con el objeto de estimular varios sentidos del usuario de forma simultánea. Algunos sistemas multimodales permiten incluso que el usuario pueda elegir entre las diversas modalidades de entrada para llevar a cabo la interacción, permitiendo así una cierta adaptación a las condiciones en las que se encuentra. Estos sistemas resultan de gran utilidad para usuarios con problemas de visión o discapacidades motoras, o entornos en los que no es aconsejable o imposible el uso de interfaces tradicionales (por ejemplo, conducción en carretera, lectura de *ebooks*, etc.) (López-Cózar, 2014).

En la actualidad, los sistemas multimodales están presentes en multitud de entornos, siendo de creciente importancia su utilización como interfaces de usuario en el acceso a servicios que se ofrecen a

través de dispositivos móviles. La creciente utilización de los dispositivos móviles, que se han convertido en una herramienta indispensable de comunicación y de trabajo en la vida de las personas, plantea la necesidad de incorporar interfaces de usuario multimodales para estos dispositivos que proporcionen una interacción más natural con el usuario aún cuando éste tenga algún tipo de impedimento.

Desde sus inicios, Google con Android ha tratado de potenciar la interacción oral entre el usuario y el dispositivo móvil a través del reconocimiento automático del habla o *ASR* (*Automatic Speech Recognition*, en inglés) para la entrada al sistema, y de la síntesis de texto a voz o *TTS* (*Text-to-Speech*, en inglés) para la salida del sistema. Dada la necesidad de contar con mecanismos multimodales de activación de servicios para dispositivos móviles anteriormente mencionada, tanto el reconocimiento automático del habla como la síntesis de texto a voz en aplicaciones Android, han evolucionado incesantemente desde su primera integración en dispositivos Android.

En cuanto al reconocimiento automático del habla en dispositivos Android, desde un principio Android incorpora la aplicación de búsqueda por voz, instalada por defecto en la mayoría de los dispositivos Android. Dicha aplicación muestra inicialmente la ventana con el texto "Hablar ahora" y consecutivamente transfiere el audio a los servidores de Google para que se haga el reconocimiento. Posteriormente, Android ha incorporado nuevas características al reconocimiento de voz en todas sus actualizaciones:

- **Android 2.1** incorpora la entrada de voz por teclado que hace posible utilizar la voz para escribir en cualquier campo de texto donde se pueda escribir texto utilizando el teclado (Ej. SMS, WhatsApp, e-mail, etc).
- Con **Android 2.2**, Google lanza las acciones de voz para Android (*Voice Actions for Android*) que permiten al usuario utilizar su voz para interactuar con su teléfono mediante comandos de voz.
- Con **Android 4.0**, se incorpora a la entrada de voz la capacidad de transcribir los resultados del reconocimiento a medida que el usuario habla, sin esperar a que éste haya terminado.
- Recientemente, la búsqueda por voz de Google se ha visto potenciada con la llegada de **Android 4.1 Jelly Bean**, que ha incorporado un amplio número de acciones mediante comandos por voz.
- Por último, la llegada de **Android 4.4 KitKat** ha dotado a los dispositivos Android de un sistema de reconocimiento de voz más preciso y ha incorporado la opción de siempre a la escucha, de tal manera que cuando el usuario dice: "OK, Google", se activa el reconocimiento de voz. Además las conversaciones ya no son unidireccionales, ya que la aplicación formula preguntas al usuario con el objetivo de devolver resultados más precisos.

Por otro lado, el sistema operativo **Android 1.6 o superior** incorpora en la mayoría de sus dispositivos un motor de síntesis de voz denominado Pico TTS o Google TTS (en las versiones más recientes), permitiendo integrar la síntesis de texto a voz a cualquier aplicación. Mediante el motor de síntesis de voz, las aplicaciones son capaces de producir voz a partir de cualquier cadena de texto, permitiendo al usuario interactuar con la aplicación sin la necesidad de leer la pantalla del dispositivo. Todo dispositivo Android incorpora por lo general un motor TTS instalado por defecto, mediante el cual las aplicaciones son capaces de leer textos en voz alta. Sin embargo, Android ofrece la posibilidad de instalar y personalizar varios motores, aunque siempre eligiendo uno como motor TTS principal.

1.2. Objetivos

En base a la importancia que tiene la utilización de interfaces multimodales en el acceso a servicios ofrecidos a través de dispositivos móviles, el objetivo principal del presente Proyecto Final de Carrera es el desarrollo de una aplicación para dispositivos móviles Android de acceso multimodal, cuya función es la de proporcionar, a través de un sistema de diálogo multimodal, información televisiva y de cine atendiendo a los gustos registrados por el usuario. Por consiguiente, el usuario puede seleccionar la modalidad con la que desea interactuar con el dispositivo de acuerdo a sus preferencias y las condiciones del ambiente de uso. De esta forma, se persigue además que la aplicación desarrollada facilite el acceso

a los servicios que ofrece a personas con problemas de visión o discapacidades motoras, o en entornos en los que no es aconsejable o resulta imposible el uso de interfaces tradicionales (por ejemplo, en conducción).

La aplicación para dispositivos móviles Android desarrollada para este proyecto permite al usuario acceder a los siguientes servicios a través de la modalidad deseada:

- Asistente televisivo: Recomienda al usuario la programación diaria de acuerdo a sus gustos.
- Recomendación de películas: Recomienda al usuario películas de acuerdo a sus gustos.
- Búsqueda de películas: Permite al usuario solicitar información de una película (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios).
- Cartelera: Permite al usuario buscar salas de cine y obtener información de la cartelera de dicho cine.

Además, con la intención de alcanzar el objetivo principal, se han definido los objetivos parciales descritos a continuación:

- Llevar a cabo una descripción detallada de los sistemas de diálogo en cuanto a concepto, arquitectura modular y aplicaciones existentes en la actualidad.
- Realizar un estudio exhaustivo de la plataforma Android y del entorno de desarrollo de sus aplicaciones.
- Realizar un estudio completo de las posibilidades que ofrece Android para el desarrollo de aplicaciones que permitan la interacción oral con el usuario, es decir, que integren el reconocimiento automático del habla para la entrada, y la síntesis de texto a voz para la salida.
- Presentar ejemplos de asistentes virtuales y otras aplicaciones de interés para dispositivos móviles Android existentes en la actualidad y similares a la aplicación que se desarrolla para este proyecto.
- Realizar un estudio de las tecnologías que se utilizan en el desarrollo de la aplicación para dispositivos móviles Android que incluya como crear, gestionar y manipular una base de datos SQLite en Android; como desarrollar una arquitectura cliente-servidor de modo que la aplicación Android pueda acceder a una base de datos MySQL de un servidor web remoto; y como extraer contenido HTML y descargar imágenes de páginas web.

1.3. Fases de desarrollo

La división del presente proyecto en fases se ha realizado aplicando los conocimientos adquiridos en la asignatura Proyectos cursada en Ingeniería de Telecomunicación. Según la teoría estudiada en dicha asignatura, las fases de desarrollo de un proyecto se pueden agrupar en tres grandes bloques: Planificación, Ejecución y Documentación.

Para facilitar la organización de las tareas del presente proyecto se emplea un *WBS* (*Work Breakdown Structure*), herramienta utilizada para organizar el trabajo de forma jerárquica y que describe los entregables y tareas que deben realizarse para un proyecto dado.

Para construir un diagrama WBS, se definen las grandes áreas de trabajo en las que se divide el proyecto, que constituyen los paquetes de trabajo a desarrollar para lograr la meta. Posteriormente, cada uno de estos paquetes de trabajo se divide en otros más pequeños hasta lograr el desglose necesario.

La Figura 1.1 muestra el diagrama WBS con las tareas del presente proyecto. Tal y como se observa, los paquetes de trabajo que constituyen el primer nivel de jerarquía de la estructura son los tres grandes bloques de tareas descritos previamente: Planificación, Ejecución y Cierre. A partir de estos tres bloques se han desarrollado las 12 tareas que componen el proyecto (tareas con índice de 1 a 12) y que constituyen el segundo y último nivel de la jerarquía.

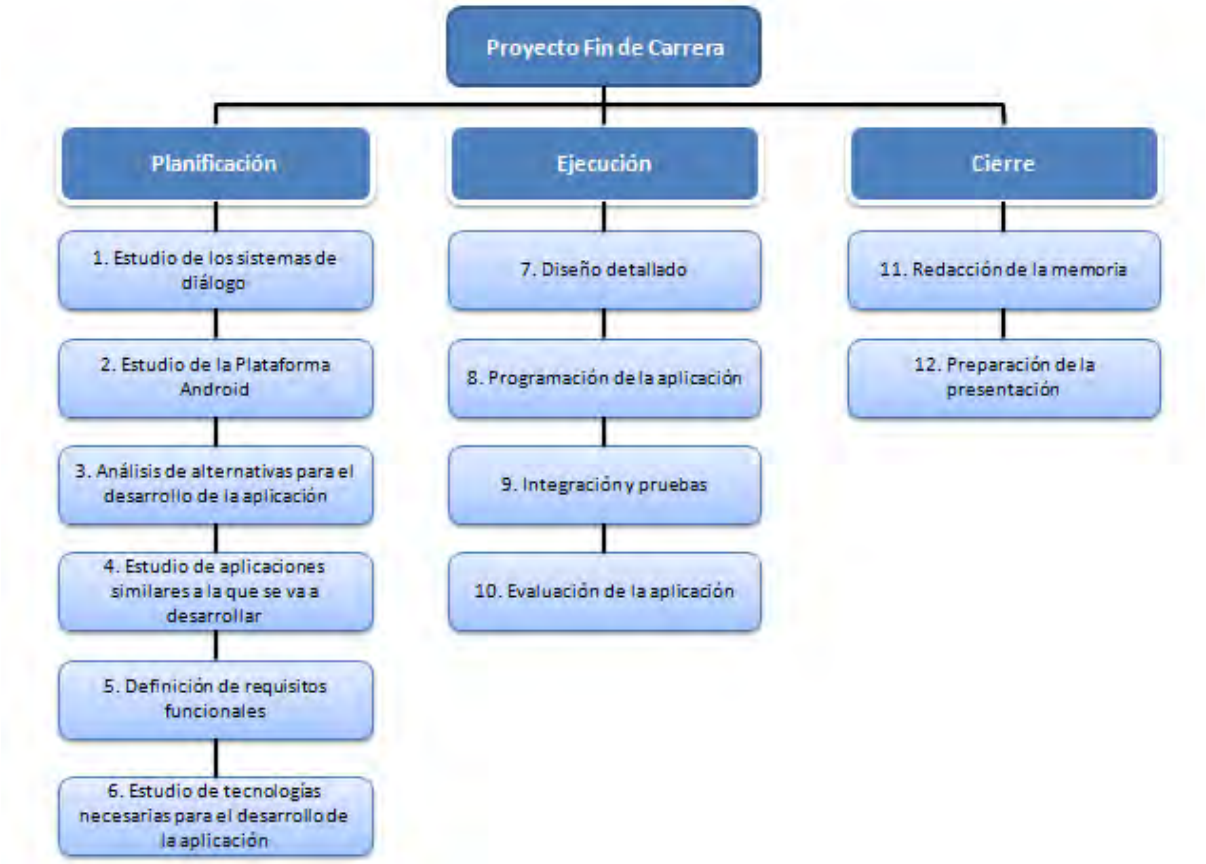


Figura 1.1: Diagrama WBS representando las tareas definidas para el proyecto

Fase 1: Planificación

- **Estudio de los sistemas de diálogo**: definición de los sistemas de diálogo y de su arquitectura, estudio de los requisitos que debe cumplir un sistema de diálogo ideal y de las limitaciones a las que están sujetos, y presentación de aplicaciones existentes en la actualidad.
- **Estudio de la plataforma Android**: realizar un estudio exhaustivo de la plataforma Android y del entorno de desarrollo de sus aplicaciones.
- **Análisis de alternativas para el desarrollo de aplicaciones para dispositivos móviles Android basadas en sistemas de diálogo**: análisis de las alternativas que ofrece Android para integrar el reconocimiento de voz y la síntesis de voz en una aplicación.
- **Estudio de aplicaciones para dispositivos móviles Android basadas en sistemas de diálogo:** presentar ejemplos concretos de asistentes virtuales y otras aplicaciones para dispositivos móviles Android basadas en sistemas de diálogo y similares a la aplicación que se quiere desarrollar para el presente proyecto.
- **Definición de los requisitos funcionales de la aplicación para dispositivos móviles Android**: definir el funcionamiento básico de la aplicación y de las necesidades que debe cubrir sin entrar en detalles.
- **Estudio de las tecnologías necesarias para desarrollar la aplicación:** estudio de los sistemas de gestión de base de datos SQLite y MySQL, del lenguaje de programación PHP, del formato para el intercambio de datos JSON, y de la implementación de algunas operaciones en

aplicaciones Android como son la ejecución de tareas en segundo plano, la extracción de contenido HTML de una página web mediante la biblioteca JSoup y la carga asíncrona de imágenes remotas.

Fase 2: Ejecución

- **Diseño detallado**: división de las distintas funcionalidades en los diferentes módulos y submódulos del sistema.
- **Programación de la aplicación**: programación de la aplicación en Android utilizando Eclipse.
- **Integración y Pruebas**: realización de pruebas funcionales para cada módulo por separado y del sistema completo hasta alcanzar una versión completamente estable.
- **Evaluación de la aplicación**: realización de las preguntas del cuestionario de evaluación de la aplicación basado en un estudio previo acerca de la evaluación de sistemas de diálogo multimodales, desarrollo de la página HTML con dicho cuestionario, recogida y análisis de resultados.

fase 3: Documentación

- **Redacción de la memoria del Proyecto Final de Carrera**.
- **Preparación de la presentación**.

La planificación temporal de las fases de desarrollo descritas en el presente apartado se realiza en la Sección 7.1 de esta memoria.

1.4. Medios Empleados

Para el desarrollo del presente Proyecto Final de Carrera se han empleado los siguientes medios:

Recursos Hardware

- Ordenador portátil.
- Smartphone HTC Desire X.
- Cable usb.
- Servidor web x10Hosting: servidor web para el almacenamiento de los ficheros PHP y de las bases de datos MySQL de la aplicación.

Recursos Software

- Entorno de desarrollo integrado de código abierto multiplataforma Eclipse.
- SDK (*Software Development Kit*) de Android: kit de desarrollo de software para Android.
- JDK (*Java development kit*): kit de desarrollo de Java.
- Plug-in ADT (Android Development Tools) para Eclipse: permite desarrollar aplicaciones Android utilizando Eclipse.
- phpMyAdmin: herramienta gratuita escrita en PHP que permite bases de datos MySQL vía web.
- Sintetizador de voz IVONA TTS.
- Aplicación de búsqueda por voz de Google.
- JSoup: librería Java que permite extraer contenido HTML de las páginas web

- LyX: herramienta gráfica multiplataforma que permite la edición de texto usando LaTeX.
- Herramienta *GanttProject* para la creación del diagrama de Gantt.
- Servicio de alojamiento de archivos multiplataforma *Dropbox*.

El coste de los medios empleados se presenta en la Sección 7.2 de esta memoria. En cuanto a la documentación que se ha empleado en la realización del presente Proyecto Final de Carrera se encuentra detallada en la sección de Bibliografía del presente documento.

1.5. Estructura de la memoria

A continuación se resume el contenido de cada capítulo del presente documento.

Capítulo 1: Introducción Incluye la motivación del proyecto, los objetivos que persigue, las fases de desarrollo, los medios empleados y la estructura de la memoria.

Capítulo 2: Estado del Arte En este capítulo se realiza un estudio detallado de los sistemas de diálogo y un análisis de las alternativas que ofrece Android para integrar el reconocimiento de voz y la síntesis de texto a voz en una aplicación. El capítulo concluye con la descripción de ejemplos concretos de asistentes virtuales y otras aplicaciones de interés para el presente proyecto.

Capítulo 3: Descripción General del sistema Este capítulo comienza con una presentación general del sistema. Para ello, se presenta la arquitectura modular del sistema y se realiza una descripción general de cada uno de sus módulos en cuanto a funcionalidad, arquitectura y tecnologías utilizadas. A continuación, se realiza un estudio completo de las tecnologías empleadas en el desarrollo del sistema y se describe la implementación de las operaciones generales.

Capítulo 4: Descripción Detallada de los módulos del sistema En este capítulo se describe de forma detallada los módulos del sistema en cuanto a su funcionalidad, arquitectura y flujo de datos. Además se presenta posibles escenarios de uso para cada módulo.

Capítulo 5: Evaluación de la aplicación Este capítulo comienza con una descripción de la metodología empleada en la evaluación de los sistemas de diálogo multimodales. Posteriormente, se realiza la evaluación de la aplicación multimodal desarrollada para dispositivos móviles Android en el presente proyecto. Para ello, se recoge a través de un cuestionario, basado en la metodología de evaluación descrita previamente, las valoraciones subjetivas de los usuarios de la aplicación.

Capítulo 6: Conclusiones y futuras líneas de trabajo En este capítulo se exponen las principales conclusiones extraídas de la realización del proyecto y las posibles líneas de trabajo que se podrían generar a partir de este proyecto con el fin de mejorar la aplicación desarrollada.

Capítulo 7: Gestión del proyecto La Gestión del proyecto incluye la planificación temporal de las tareas en las que se divide el proyecto y un análisis de los costes del proyecto, que incluye el coste de personal y del material que ha sido necesario para llevarlo a cabo.

Glosario En este apartado se recogen los principales términos utilizados en el presente documento y se acompañan de su respectiva definición o explicación con el objetivo de facilitar su comprensión al lector.

Bibliografía En este apartado se listan las referencias bibliográficas que se han consultado para la realización tanto del proyecto y de la memoria.

Capítulo 2

ESTADO DEL ARTE

El presente capítulo tiene como principal finalidad describir las posibilidades que ofrece Android para desarrollar aplicaciones que permitan la interacción oral con el usuario, es decir, que integren el reconocimiento automático del habla o *ASR* (*Automatic Speech Recognition*, en inglés) para la entrada, y la síntesis de texto a voz o *TTS* (*Text-to-Speech*, en inglés) para la salida. Estas aplicaciones son de gran utilidad para usuarios con problemas de visión o discapacidades motoras, o entornos en los que no es aconsejable o imposible el uso de interfaces tradicionales (por ejemplo, conducción en carretera, lectura de *ebooks*, etc.).

En primer lugar, se realiza una breve descripción de los sistemas de diálogo, de su arquitectura y módulos, y de sus aplicaciones. A continuación, se estudia las alternativas que ofrece Android para integrar el reconocimiento de voz en una aplicación. Para ello, se comienza con una introducción en la que se describe la evolución que ha tenido el reconocimiento de voz en Android desde sus inicios hasta la actualidad y consecutivamente, se realiza un estudio detallado sobre la integración del reconocimiento de voz en una aplicación Android.

Posteriormente, se realiza un estudio sobre las alternativas que ofrece Android para integrar la síntesis de texto a voz en una aplicación. Para ello, se realiza una introducción de la síntesis de texto a voz en Android, que incluye una descripción detallada del motor de síntesis que viene instalado por defecto en la mayoría de los dispositivos Android y a continuación, se procede a explicar los conceptos relacionados con la integración de la síntesis de texto a voz en una aplicación Android. Finalmente, se realiza un estudio sobre las posibilidades que ofrece Android como sintetizadores de texto a voz con el objetivo de seleccionar aquel que mejor se integre a una aplicación y mejores prestaciones ofrezca.

Se concluye el capítulo describiendo algunos ejemplos de asistentes virtuales y otras aplicaciones de interés para el presente proyecto.

2.1. Los sistemas de diálogo hablado

2.1.1. Introducción a los sistemas de diálogo hablado

Los sistemas de diálogo hablado (*spoken dialogue systems*) son sistemas informáticos que reciben como entrada frases del lenguaje natural expresadas de forma oral y generan como salida frases del lenguaje natural expresadas asimismo de forma oral. La finalidad de estos sistemas es emular el comportamiento inteligente de un ser humano que realiza una tarea concreta (López-Cózar, 2014).

Los sistemas de diálogo hablado intentan facilitar la interacción natural mediante el habla entre un usuario y un sistema informático, liberándole de utilizar los interfaces tradicionales como el teclado, el ratón o la pantalla. Por ello, estos sistemas utilizan usualmente una secuencia de interacciones entre la persona y la máquina que permiten que el objetivo del usuario se vaya alcanzando gradualmente a través de varios turnos de diálogo (Griol, 2007).

La Figura 2.1 (Griol, 2007) muestra las acciones básicas que realiza un sistema de diálogo. Como se aprecia, el sistema genera un mensaje inicial que sirve habitualmente para dar la bienvenida o informar

al usuario sobre las características del sistema. Tras cada intervención del usuario, el sistema realiza un conjunto de acciones básicas que se repiten cíclicamente como respuesta a cada acción del usuario:

- Reconoce la secuencia de palabras emitidas por el usuario.
- Extrae el significado de las palabras reconocidas, es decir, comprende qué información es útil en el dominio del sistema.
- Realiza operaciones de acceso a la base de datos u otros recursos del sistema, en los que se almacena la información que solicita el usuario o se registran las operaciones que desea conocer.
- Decide qué acción o acciones deben realizarse a continuación de cada solicitud del usuario, es decir, qué respuesta debe suministrar el sistema.
- Reproduce un mensaje hablado que informa al usuario sobre la acción seleccionada por el sistema.

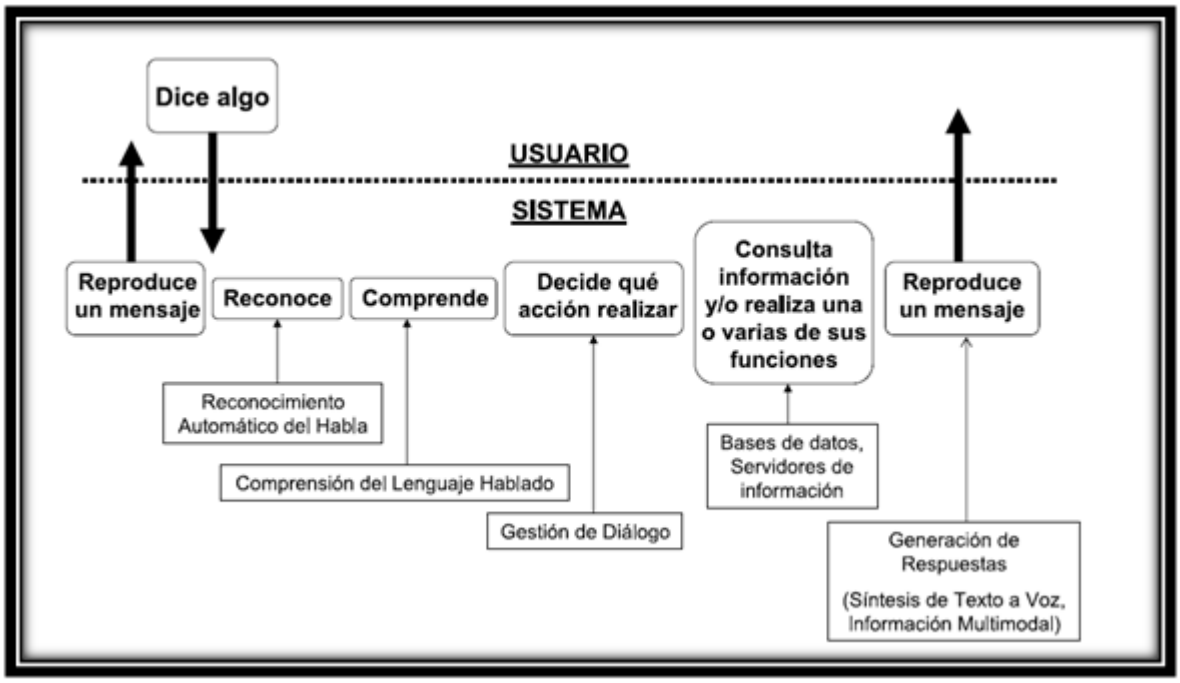


Figura 2.1: Diagrama de acciones de un sistema de diálogo

Debido al gran número de operaciones que deben realizarse, todo sistema de diálogo hablado se descompone en un conjunto módulos con el fin de de desglosar las dificultades entre los diferentes componentes del sistema. La arquitectura modular de un sistema de diálogo se estudiará en la Sección 2.1.2.

Un sistema de diálogo ideal debe cumplir una serie de requisitos (Llisterri, 2006):

- Reconocer sin dificultades el habla espontánea.
- Comprender enunciados sin restricciones de contenido.
- Proporcionar respuestas con sentido, gramaticalmente bien formadas y pragmáticamente adecuadas.
- Responder con voz completamente natural.

- Permitir una interacción multimodall[1] a través del habla y de la imagen.

No obstante, a pesar de que la investigación ha avanzado notablemente, construir sistemas de diálogo ideales constituye un reto por los motivos que se resumen a continuación (Llisterri, 2006).

- Los sistemas de diálogo están sujetos a las limitaciones del reconocimiento del habla. No se dispone aún de reconocedores que puedan procesar con un cien por cien de fiabilidad la variación propia del habla completamente espontánea, la multiplicidad de realizaciones fonéticas de locutores de diversa procedencia geográfica y social, y las interferencias producidas por el entorno cuando el sistema tiene que utilizarse, por ejemplo, dentro de un vehículo en marcha o hablando por un teléfono móvil en plena calle.

- Aun suponiendo que se pudiese contar con los mejores reconocedores, un sistema de diálogo debería comprender cualquier enunciado, con independencia del tema o de su complejidad sintáctica, semántica y pragmática. Sin embargo, la comprensión del lenguaje, es decir, la creación automática de una representación del contenido de un enunciado a partir de los resultados del reconocedor de habla, está limitada a lo que se denominan dominios restringidos, ámbitos muy concretos a los que se aplica un determinado sistema de diálogo, como pueda ser la información de horarios de trenes, la reserva de billetes de avión o la información ciudadana.

- Las respuestas de un sistemas de diálogo ideal deben confundirse con las de un humano, no sólo siendo gramaticalmente adecuadas, sino que además deben abarcar cualquier tema y presentar las mismas características en cuanto a riqueza y flexibilidad que las del lenguaje natural. Para llegar a estos resultados debería disponerse de un sofisticado sistema de generación del lenguaje capaz de emular los mecanismos mediante los que las personas consiguen producir enunciados que responden acertadamente a cualquier situación, algo que todavía no está al alcance de la tecnología actual.

- En un sistema de diálogo ideal no es preciso que el interlocutor humano guíe al sistema confirmando que la interpretación de sus preguntas es la correcta, contrariamente a algunos de los sistemas actuales de diálogo.

- Los métodos de conversión de texto en habla, aún no permiten crear una voz sintetizada que pueda confundirse con la de una persona ni son capaces, por el momento, de ofrecer una realización prosódica de los enunciados idéntica a la del habla natural. Este es uno de los principales motivos por los que un usuario se percata inmediatamente de que, al llamar a un determinado servicio, quien le responde no es un interlocutor humano, sino un sistema de diálogo.

Aun así, a pesar de las limitaciones a las que se han visto enfrentados los sistemas de diálogo a lo largo de su historia, el incesante avance en la investigación de tecnologías del habla ha permitido que en la actualidad los sistemas de diálogo sean factibles y estén presentes en multitud de entornos: sistemas que proporcionan información sobre horarios y precios de transportes públicos, sistemas de información meteorológica, servicios de banca electrónica, aplicaciones accesibles desde los vehículos, sistemas que facilitan el acceso a la información a personas con discapacidades, acceso a servicios y control de máquinas vía telefónica (fija o móvil), portales de voz, etc (Griol, 2007). En la Sección 2.1.3 se describirá con más detalle las aplicaciones de los sistemas de diálogo.

2.1.2. Arquitectura de un sistema de diálogo

Tal y como se ha descrito en la sección anterior, los sistemas de diálogo constan de un gran número de operaciones. Como consecuencia, todo sistema de diálogo se descompone en diferentes módulos con el propósito de desglosar las dificultades entre los diferentes componentes que componen el sistema.

[1]En una interacción multimodal el usuario no está restringido a utilizar el habla como único canal de comunicación, sino que puede utilizar varios dispositivos de entrada, como por ejemplo un teclado, un ratón, un micrófono, una cámara, una pantalla sensible al tacto, una PDA, etc. Asimismo, el sistema multimodal puede utilizar diversos canales de salida para proporcionar información al usuario como por ejemplo, voz, texto, gráficos o imágenes (López-Cózar, 2014).

La Figura 2.2 (Griol, 2007) muestra la arquitectura modular descrita para el desarrollo de sistemas de diálogo hablado y a continuación, se realiza la descripción de los módulos que forman parte de dicha arquitectura (Griol, 2007), (Corrales, 2010).

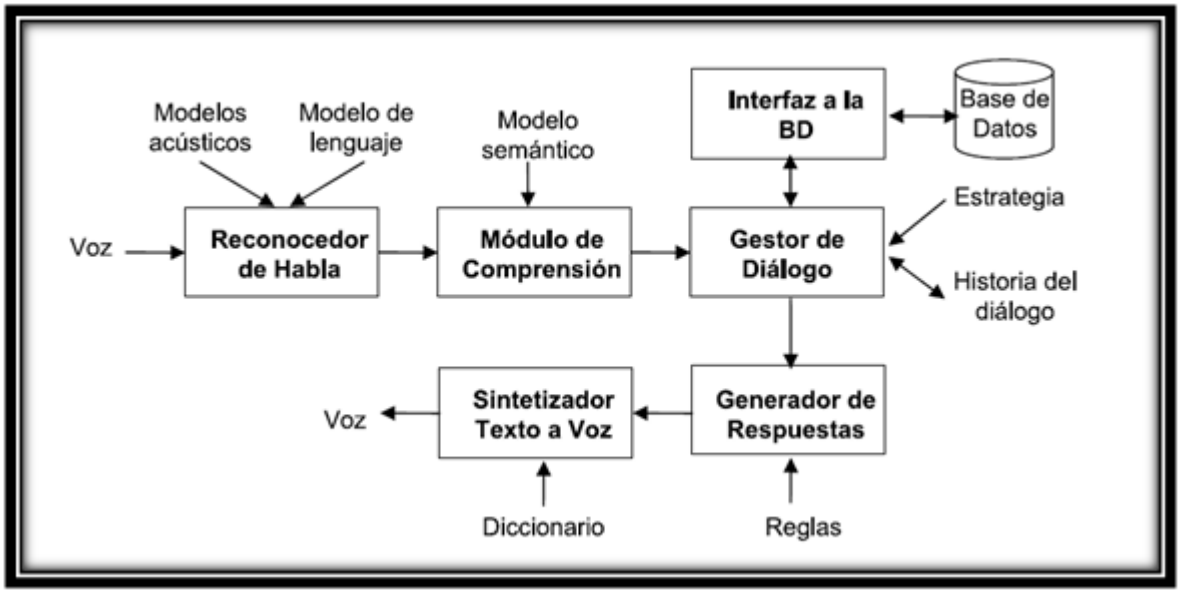


Figura 2.2: Arquitectura modular de un sistema de diálogo hablado

Módulo de Reconocimiento Automático del Habla (RAH)

El módulo de Reconocimiento Automático del Habla (RAH) se encarga de reconocer la señal de voz emitida por el usuario y devolver la(s) secuencia(s) de palabras reconocida(s) más probable mediante la aplicación de técnicas de proceso de señal de voz. Se compone de una etapa de preproceso (modelado de la señal), una etapa Acústico-Fonética (modelado acústico de unidades subléxicas y/o léxicas), y una etapa Sintáctico-Semántica (modelado del lenguaje).

Las diferencias lingüísticas y de pronunciación entre diferentes locutores producen una variabilidad en el resultado del RAH, lo que conlleva a errores de comunicación impredecibles. Para reducir dichos errores, en el desarrollo de este módulo se combinan un amplio número de disciplinas: Fisiología, Acústica, Procesamiento de Señales, Inteligencia Artificial, y Ciencia de la Computación. Además de errores procedentes de la variabilidad antes comentada, el sistema está expuesto a errores ambientales y procedentes de los canales o dispositivos de comunicación, por lo que los expertos están obligados a someter al sistema a cierta robustez que logre eliminar todos los errores.

Módulo de Comprensión del Habla / Módulo de Procesamiento del Lenguaje Natural (PLN)

El módulo de Procesamiento del Lenguaje Natural (PLN) es el encargado de obtener una representación semántica del significado de las(s) secuencia(s) de palabras reconocida(s) en el módulo RAH. Puede entenderse como un cambio en el lenguaje de representación, de lenguaje natural a un lenguaje semántico, de forma que se mantenga el significado del mensaje.

El PLN se realiza a través de los siguientes análisis realizados de forma secuencial:

- Análisis morfológico-léxico: Divide las palabras en lexemas (parte de las palabras que indica su semántica) y en morfemas (infijos y sufijos mediante los cuales se obtienen las distintas clases de palabras).
- Análisis sintáctico: Se obtiene la estructura jerárquica de las frases, y se extrae el significado de la estructura sintáctica compleja obtenida a partir del significado de sus constituyentes.

- Análisis semántico: Se obtiene un procesamiento del significado.
- Análisis pragmático: Añade información al análisis semántico de la frase en función del contexto en el que aparece.

Además, se pueden añadir otros niveles del conocimiento como es la información fonológica (relación de las palabras y su sonido al pronunciarlas), análisis del discurso (estudio de información previa que puede interferir en el significado de la información actual), y lo que se denomina como "conocimiento del mundo" (información esencial sobre la estructura del mundo necesaria para mantener una conversación).

Gestor de Diálogo(GD)

El Gestor de Diálogo (GD) se considera el "cerebro" del sistema de diálogo ya que es el encargado de decidir qué paso debe dar el sistema tras cada intervención del usuario. Para ello, se apoya en la interpretación semántica de la petición del usuario, la historia del proceso de diálogo, la información de la aplicación disponible en ese punto y el estado del sistema.

El GD debe cumplir las siguientes funciones:

- Actualizar el contexto del diálogo.
- Proveer de contexto en el que basar las interpretaciones.
- Coordinar el resto de módulos del sistema.
- Decidir qué información dar al usuario, y cuándo hacerlo.

Módulo de Consulta a la Base de Datos de la Aplicación

El módulo de Consulta a la Base de Datos de la Aplicación es el encargado de recibir peticiones de consulta a la base de datos por parte del gestor de diálogo, procesarlas y devolver el resultado al gestor.

Módulo de Generación de Respuestas / Módulo de Generación del Lenguaje Natural (GLN)

El módulo de Generación del Lenguaje Natural (GLN) es el encargado de recibir la respuesta del sistema en forma de cierta representación formal y tiene como función la generación de una frase, gramaticalmente correcta y en un lenguaje lo más cercano posible al lenguaje natural, que transmita el mensaje generado por el gestor de diálogo. La respuesta del sistema proporcionada por el GLN puede incorporar otras modalidades de información (vídeo, tablas con datos, gestos a reproducir por un avatar, etc.).

Sintetizador de Texto a Voz

El Sintetizador de Texto a Voz transforma la respuesta que recibe del sistema como texto en lenguaje natural, en la señal de audio correspondiente que recibirá el usuario.

Los sintetizadores de texto a voz se componen de dos partes fundamentales:

- *Front-end*: Se encarga de convertir el texto plano, compuesto por símbolos y abreviaturas, en sus palabras asociadas, o lo que es lo mismo, normaliza el texto y asigna transcripciones fonéticas a las palabras dividiendo y marcando el texto en frases, cláusulas y oraciones.
- *Back-end*: Se encarga de convertir la representación lingüística simbólica en sonidos, utilizando diversas técnicas para simular los sonidos producidos por las cuerdas vocales humanas.

2.1.3. Aplicaciones de los sistemas de diálogo

La Tabla 2.1 (Griol et al., 2009), (Llisterri, 2006), (García, 2011) , resume algunos ejemplos de aplicaciones de los sistemas de diálogo.

Función	Tarea desarrollada	Ejemplos
Gestión de Información	Gestión de información, como en el caso de los servicios de atención al cliente que orientan en la solución de un problema, el de las centralitas automáticas que ponen en contacto al usuario con la persona adecuada, etc. Algunos sistemas permiten gestionar datos pertenecientes al propio usuario (Sistemas que permiten la consulta del correo electrónico a través del teléfono).	*Verbio Technologies* ("Verbio", 2014): Servicios de atención al cliente. España. *AthosMail* (Turunen et al., 2004): Sistema multilingüe y adaptativo para la lectura de correos electrónicos utilizando teléfonos móviles. EE.UU.
Consulta de información	Se accede a datos almacenados en un repositorio de información como puede ser una página web o una base de datos. Se utilizan en muchos ámbitos: proporcionar horarios y precios de transportes, información meteorológica, información ciudadana sobre horarios de servicios, sobre trámites en la administración o sobre la oferta cultural, etc.	*Júpiter* ("Jupiter", 2013): Sistema de información meteorológica por teléfono. EE.UU. *Dihana* (Griol et al., 2008): Información de viajes en tren. España. *Pegasus* ("Pegasus", 2013): Información de viajes en avión. EE.UU.
Reservas	Además de proporcionar información, realizan tareas algo más complejas, como son reservas a través de la aplicación de voz.	*Mercury* ("Mercury", 2013): Proporciona información sobre vuelos y permite hacer reservas. EE.UU.
Transacciones	Relacionadas con el comercio electrónico (venta de entradas o de billetes de tren y de avión) y con la banca electrónica.	*Voice Ticketing* ("Voice Ticketing", 2011): Automatización del proceso completo de compra de entradas, billetes de transporte, invitaciones para espectáculos, eventos o viajes. *Natural Vox* ("Natural vox", 2011): Permite comprar un billete de autobús, pedir cita con el médico, realizar una transferencia, solicitar un presupuesto para el seguro del coche, etc. España.
Control remoto	La interacción oral también es útil para el control de dispositivos, especialmente en entornos inteligentes.	MIMUS (G. Pérez et al., 2006): Sistema de diálogo multimodal para controlar un hogar inteligente. España.
Tutorización	Los sistemas de diálogo se emplean también en el ámbito de la educación, particularmente con el fin de mejorar habilidades fonéticas y lingüísticas de los usuarios.	LISTEN (Mostow, 2008): Tutor de lectura. España. VOCALIZA (Vaquero et al., 2006): Sistema para terapias de voz. España. ITSPOKE (Litman y Silliman, 2004): Tutor para dar información y corregir errores de aprendizaje. EE.UU.
Robótica	Integración de los sistemas de diálogo en la robótica.	Cogniron (Menezes et al., 2007): Robot cognitivo. Europa.
Compañeros virtuales	Los agentes virtuales y compañeros constituyen la aplicación más compleja de los sistemas de diálogo.	Companions (Benyon y Mival, 2007): Compañero personalizado multimodal. Europa.

Tabla 2.1: Aplicaciones de los sistemas de diálogo

Las aplicaciones de los sistemas de diálogo de la Tabla 2.1 están clasificadas según su utilidad y tareas que realizan, y aparecen ordenadas de menor a mayor nivel de complejidad. Tal y como se observa, el número de entornos y tareas en los que pueden aplicarse los sistemas de diálogo es muy amplio.

2.2. Reconocimiento Automático del Habla en Android

2.2.1. Introducción al reconocimiento automático del habla en Android

Desde sus inicios, Google con Android ha tratado de potenciar la forma de interactuar con el sistema a través del reconocimiento de voz, el cual permite procesar la señal de voz emitida por el usuario y reconocer la información contenida en ésta, convirtiéndola en texto.

Desde un principio, Android incorporó la aplicación de búsqueda por voz que viene instalada por defecto en la mayoría de los dispositivos Android. Dicha aplicación muestra inicialmente la ventana con el texto "Hablar ahora" y consecutivamente transfiere el audio a los servidores de Google para que se haga el reconocimiento. Al estar parte del proceso de reconocimiento de voz en los servidores de Google, es necesario tener acceso a Internet para que la aplicación de búsqueda por voz de Google pueda funcionar.

Posteriormente, **Android 2.1** incorporó la entrada de voz por teclado con lo que fue posible usar la voz para escribir en cualquier campo de texto donde se pudiera escribir texto usando el teclado (Ej. SMS, WhatsApp, e-mail, etc).

Con **Android 2.2**, Google lanzó *Voice Actions for Android* que permitieron al usuario utilizar su voz para interactuar con su teléfono en al menos 10 nuevas formas.

Posteriormente, con **Android 4.0** se incorporó a la entrada de voz la capacidad de transcribir los resultados del reconocimiento a medida que el usuario habla, sin esperar a que éste haya terminado.

Recientemente, la búsqueda por voz de Google se ha visto potenciada con la llegada de **Android 4.1 Jelly Bean**, que permite realizar un amplio número de acciones mediante comandos por voz. Además, en la actualización de la aplicación de búsqueda por voz, se ha habilitado el reconocimiento de voz *offline*, lo que permite utilizar el reconocimiento de voz sin la necesidad de tener acceso a Internet. No obstante, no existe ninguna API disponible para implementar esta funcionalidad siendo únicamente necesario activarla desde la opción del menú Ajustes->Idioma y teclado->Búsqueda por voz->Reconocimiento de voz sin conexión, que permite escoger de forma individual los idiomas de los que se quiere hacer utilización del reconocimiento de voz *offline*. El problema de esta funcionalidad es que no se encuentra disponible en todos los dispositivos y no existe ninguna documentación que haga referencia a esta funcionalidad y a que dispositivos se puede aplicar.

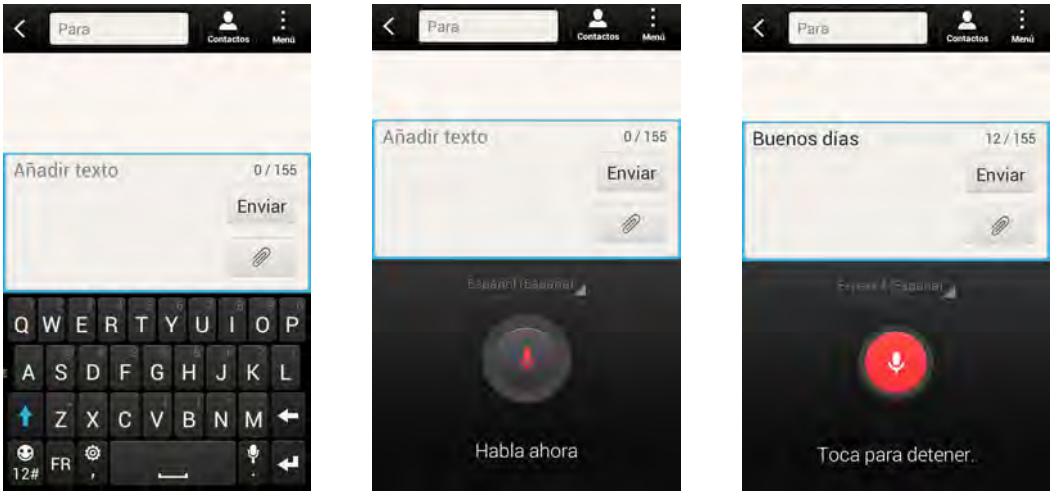
Por último, la llegada de **Android 4.4 KitKat** ha dotado a los dispositivos Android de un sistema de reconocimiento de voz más preciso y ha incorporado la opción de siempre a la escucha, de tal manera que cuando el usuario dice: "OK, Google", se activa el reconocimiento de voz. Además las conversaciones ya no son unidireccionales, ya que la aplicación formula preguntas al usuario con el objetivo de devolver resultados más precisos.

2.2.1.1. Entrada de voz por teclado

La entrada de voz por teclado (Gruenstein, 2010), (Google Nexus One, 2010) se incorpora a través de un botón de un micrófono que al pulsarse, permite ingresar texto mediante la voz en cualquier aplicación con campos de texto, excepto en los campos para contraseña. Hay que tener en cuenta que la entrada de voz es una función que usa el servicio de reconocimiento de voz de Google por lo que para que funcione, es necesario tener acceso a Internet ya que el reconocimiento de voz se hace desde los servidores de Google. Además, la aplicación de búsqueda por voz de Google debe estar instalada en el dispositivo.

La Figura 2.3 ilustra el proceso de introducción de texto mediante voz. En primer lugar, se pulsa el campo de texto donde se quiere introducir el texto y se pulsa la tecla del micrófono en el teclado en pantalla (Figura 2.3.a). A continuación aparece en pantalla "Hablar ahora" y se puede comenzar a decir las palabras que quieren ser introducidas en el campo de texto (Figura 2.3.b). Cuando el usuario

deje de hablar, el servicio de reconocimiento de voz transcribirá lo que haya dicho y lo ingresará en el campo de texto (Figura 2.3.c). Cabe destacar que para dispositivos Android 4.0 o superior, los resultados del reconocimiento se transcriben en el campo de texto a medida que el usuario habla, sin esperar que éste haya terminado.



(a) El usuario pulsa la tecla del micrófono

(b) El reconocedor se mantiene a la espera de la entrada por voz

(c) Resultados de la entrada por voz

Figura 2.3: Introducción de un texto por voz

Para entrar en el menú de configuración del dictado por voz de Google, el usuario debe dirigirse a Ajustes->Idioma y Teclado->Dictado por voz de Google. Tal y como se observa en la Figura 2.4, dicho menú consta de dos opciones:

- Idioma: Permite seleccionar el idioma de la entrada por voz. El idioma seleccionado para la entrada por voz puede ser diferente del idioma utilizado en la interfaz del teléfono. Google incorpora regularmente nuevos idiomas.
- Bloquear palabras ofensivas: Al seleccionar esta opción se bloquean los resultados de voz que contengan lenguaje ofensivo.

Figura 2.4: Ajustes de dictado por voz de Google

2.2.1.2. Evolución de las acciones de Voz para Android

Android 2.2 integró el reconocimiento de voz a todo el sistema, permitiendo al usuario interactuar con su dispositivo mediante *Voice actions for Android* (Barra, 2010), (Google Mobile, 2010), inicialmente disponibles en Inglés. Un año después, Google lanzó una actualización interna en su aplicación de búsqueda por voz para que las acciones por voz se pudiesen utilizar en otros idiomas, entre ellos en español con las Acciones de voz para Android (Google España, 2011), (A. Pérez, 2011). Para utilizar las acciones de voz se pulsa o bien el icono de la aplicación de búsqueda por voz o bien el micrófono del *widget* de búsqueda de Google.

La Tabla 2.2 (Google España, 2011), (A. Pérez, 2011) muestra las acciones que incorporó Android 2.2 para el idioma español.

Acción de voz	Comando de voz	Ejemplo
Enviar mensajes	Enviar mensaje a [contacto] [mensaje]	Enviar mensaje a Carlos Cuesta, Voy a llegar tarde. Llegare a casa a las 9.
Llamar a contactos	Llamar a [nombre] (y opcionalmente a que teléfono: móvil, casa, trabajo)	Llamar a María Pérez en casa.
Llamar a empresas	Llamar a [nombre de la empresa] (y opcionalmente la ubicación)	Llamar a Little Paco Barcelona.
Mostrar mapa en *Google Maps*	Mapa de [dirección/ciudad]	Mapa de Londres.
Como llegar en *Google Maps*	Cómo llegar a [dirección/ciudad/nombre de la empresa]	Cómo llegar a calle Mayor 27.
Visitar una página Web	Ir a la pagina web de [sitio web]	Ir la pagina web de Wikipedia.
Iniciar ruta con *Google Navigation*	Guíame a [destino]	Guíame a la Sagrada familia.
Búsqueda en Google	[Consulta]	Imágenes de la Ciudad de las Ciencias al atardecer.

Tabla 2.2: Acciones de voz para Android 2.2 para el idioma español

Estableciendo el idioma del dispositivo en inglés (i.e. Seleccionando la opción en Ajustes->Idioma y teclado -> Idioma del sistema -> *English*), las acciones de voz para Android 2.2 permitieron realizar además de las acciones ya descritas las listadas a continuación (Barra, 2010), (Google Mobile, 2010).

- Escribir una nota: *note to self [note].*
- Escuchar música: *Listen to [artist/song/album].* Se necesita tener instalada alguna aplicación de música (Ej. Pandora, Last.fm, Rdio, mSpot).
- Enviar un email: *Send email to [contact] [message].*

Las acciones de voz han evolucionado rápidamente con cada nueva versión de Android. Con las acciones de voz de Android 2.2 se logró realizar la traducción de voz a texto en tiempo real, enviar comandos con identificadores específicos y que requerían una lógica compleja por parte de los servidores de Google para enviar una respuesta fiable, y el que se pudiese utilizar el servicio por la mayor parte de los usuarios con cierta precisión. Siempre y cuando se dispusiese de una conexión a Internet y que el usuario supiese cuales eran los comandos de voz soportados, las acciones de voz de Google de Android 2.2 han funcionado correctamente.

Recientemente, se ha mejorado la aplicación de búsqueda por voz de Google con la llegada de **Android 4.1 Jelly Bean**. Dicha actualización de la búsqueda por voz resulta ser una expansión de las acciones de voz para Android, existentes desde el 2010. Dicha evolución de las acciones de voz de Google se deben principalmente a dos proyectos de Google (Holly, 2012):

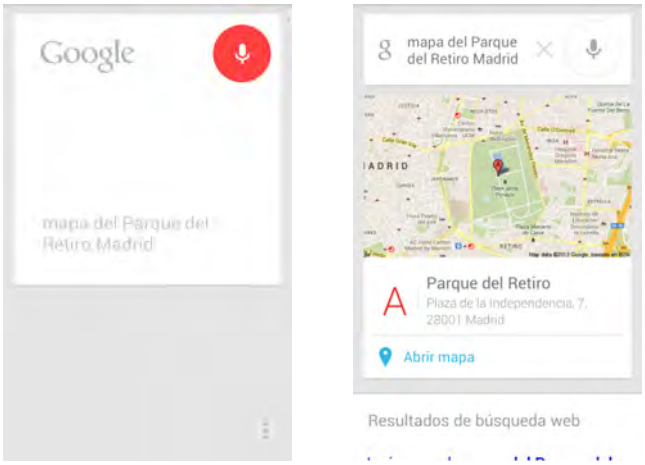
- *Project Majel* (Shanklin, 2011): El objetivo del proyecto Majel ha sido convertir la aplicación de búsqueda por voz en un asistente personal parecido al de Siri de Apple ("Siri", 2014), capaz de de interpretar un lenguaje totalmente cotidiano y de responder preguntas muy complejas.

- *Google's Knowledge Graph project* (Cardinal, 2012): El objetivo del proyecto Google's Knowledge Graph ha sido mejorar el motor de búsqueda de Google de manera que se responda al usuario en la misma página de resultados de búsqueda a través de una tarjeta.

La nueva actualización de la aplicación de búsqueda por voz permite al usuario controlar su teléfono o *tablet* mediante un amplio abanico de comandos de voz (Ej. Enviar SMS, establecer alarmas y recordatorios, etc.). Además, de forma similar a Siri, a través de la aplicación de búsqueda por voz, el usuario puede preguntar al dispositivo y que este responda a la pregunta mediante la de voz. La respuesta a la pregunta formulada por el usuario se presenta también de forma escrita en una tarjeta que contiene la información relacionada con la respuesta a pregunta y que se puede actualizar según los requisitos del usuario[2]. Al mover dicha tarjeta, se visualizan los resultados de búsqueda de Google.

Durante el congreso de Google I/O 2013 ("Google I/O", 2013), se presentó la funcionalidad de la nueva actualización de la aplicación de búsqueda por voz, resaltando sus fortalezas. En primer lugar, la nueva actualización de la aplicación de búsqueda por voz es capaz de interpretar y entender un lenguaje totalmente cotidiano, con palabras complejas e inusuales. Por lo tanto, no requiere que el usuario memorice comandos específicos. Además, incorpora las acciones de voz el funcionamiento del sistema en modo *offline*, que permite utilizar el dictado por voz de Google aunque no se disponga de una conexión a Internet.

La Figura 2.5 muestra la nueva aplicación de búsqueda por voz de Google para Android Jelly Bean. Tal y como se observa, el usuario pide que se muestre el mapa del Retiro, y la aplicación de búsqueda por voz lo muestra mediante una tarjeta.



(a) Reconocimiento de voz

(b) Resultados de la búsqueda por voz

Figura 2.5: Aplicación de Búsqueda por voz de Google

2.2.1.3. Utilización de los comandos de acciones de voz y de la búsqueda por voz en dispositivos Android 4.1 Jelly Bean

Para iniciar una acción o una búsqueda utilizando la voz, se pulsa el icono de micrófono en el escritorio o en la aplicación de Búsqueda de Google. Si el dispositivo utiliza Android 4.1 o versiones posteriores, no es necesario pulsar el micrófono, basta con abrir la aplicación de Búsqueda de Google

[2]La tarjetas son configurables. Tienen un pequeño icono de información en la esquina superior derecha que, al tocarlo, despliega un breve menú de configuración de la tarjeta en cuestión.

y decir “Google”[3]. La capacidad de iniciar una búsqueda o una acción al decir “Google” se denomina detección de palabras clave. Para activar o desactivar esta función, se abre la aplicación de Búsqueda de Google, y se va al menú y toca Ajustes -> Voz -> Detección de palabras clave (“Cómo usar tu voz en Android”, 2014).

Búsquedas por voz en dispositivos Android 4.1 Jelly Bean

Después de tocar el icono de micrófono o de decir “Google”, Google escucha lo que dice el usuario e inicia la búsqueda o la acción que describe. Si la búsqueda por voz no ha comprendido lo dicho por el usuario, se muestra una lista con posibles opciones para que el usuario seleccione una entre ellas.

Es posible que Google responda al usuario con una respuesta hablada. Entre las respuestas que se pueden obtener se incluyen hechos, condiciones meteorológicas, cotizaciones de bolsa, estados de vuelos, resultados deportivos, conversiones monetarias o cálculos matemáticos.

La Tabla 2.3 (Raphael, 2012), (“Cómo usar tu voz en Android”, 2014) agrupa en distintas categorías algunos ejemplos de las consultas por voz que permite realizar la nueva actualización de la aplicación de búsqueda por voz. No funcionan correctamente todos los comandos en español, pero poco a poco se irán habilitando. La Figura 2.6 muestra algunas capturas de pantalla asociadas a las respuestas de los ejemplos de la Tabla 2.3.

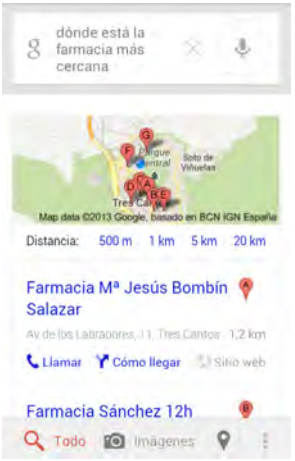
Categoría	Descripción	Ejemplo
Tiempo	Permite solicitar la previsión del tiempo en la ubicación actual, en casa, en el trabajo, etc.	- ¿Va a llover este fin de semana? (Figura 2.6.a) - ¿Qué tiempo hace en Valencia? -¿Qué tiempo va a hacer mañana por la mañana?
Sitios (i.e. Restaurantes y negocios)	Permite solicitar información acerca de los restaurantes y negocios que se encuentran en una ubicación determinada. Si existen varios resultados se muestra el mapa de la zona solicitada con los diferentes resultados. El usuario puede pulsar cualquiera de las ubicaciones encontradas para solicitar información detallada acerca del restaurante o negocio.	-¿Cual es el restaurante chino más cercano? -¿Dónde está la farmacia más cercana? (Figura 2.6.b)
Deportes	Permite solicitar información acerca de resultados de partidos ya jugados o acerca de horarios de próximos partidos.	-¿Cuándo es el partido del Barça? -¿Ganó el Real Madrid el partido de anoche?
Horas y Fechas de Eventos	Permite dar información relacionada con horas y fechas de eventos.	- Hora: ¿Qué hora es en Londres? (Figura 2.6.c) - Eventos: ¿A qué hora se pone el sol?
Estado de vuelos.	Permite solicitar información acerca del estado de un vuelo.	-¿Cuándo sale el vuelo 900 de United Airlinest?
Finanzas.	Permite solicitar información acerca de la cotización en bolsa.	-¿Cuál es la situación del S&P 500 hoy?
Conversiones y cálculos.	Permite solicitar el resultado de cualquier cálculo o conversión.	-¿Cuánto es 37 por 7? (Figura 2.6.d) -¿Cuántos euros son 78 dólares? -¿Cuántos Kilómetros hay en una milla?

[3]Para detectar cuándo el usuario dice "Google" para iniciar una búsqueda por voz o una acción de voz, Google analiza el sonido registrado por el micrófono del dispositivo a intervalos inferiores a pocos segundos. El sonido se descarta inmediatamente después de analizarlo y no se almacena en el dispositivo ni se envía a Google.

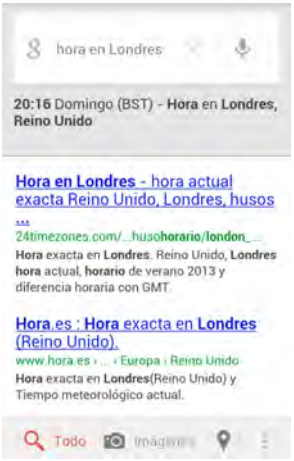
Categoría	Descripción	Ejemplo
Curiosidades	Permite solicitar información acerca de un amplio abanico de temas (Ej. Celebridades, cine, literatura, geografía, etc.)	-¿Quien fundó PayPal? -¿Quien dirigió el Caballero Oscuro? -¿Cual es el reparto de Prison Break? -¿Qué edad tiene Robert de Niro? (Figura 2.5.e) -¿Cuando murió John Lenon? -¿Quien escribió los Pilares de la tierra? -¿Cuánto dura Titanic? -¿Cual es la canción de Friends?
Imágenes	Permite al usuario solicitar imágenes.	-Ver imágenes del puente Golden Gate
Traducciones	Permite traducir cualquier palabra a cualquier idioma.	-¿Cómo se dice pepino en inglés? (Figura 2.5.f)

Tabla 2.3: Posibilidades ofrecidas por la aplicación de Búsqueda por voz de Google para Android 4.1

(a) Tarjeta de previsión del tiempo



(b) Tarjeta de búsqueda de sitios



(c) Tarjeta con información de la hora en Londres

(d) Tarjeta de Cálculos

(e) Tarjeta con información de celebridades

(f) Trajeta con la traducción de una palabra

Figura 2.6: Respuestas a las búsquedas por voz para Android 4.1

Acciones de voz en dispositivos Android 4.1 Jelly Bean

Las acciones de voz[4] permiten al usuario controlar el teléfono a través de comandos de voz. Después de tocar el icono de micrófono o de decir "Google", el usuario dice la acción de voz que quiere utilizar. También puede decir "ayuda" para ver ejemplos del tipo de cosas que puede decirle a Google para que las busque.

La Tabla 2.4 (Raphael, 2012), ("Cómo usar tu voz en Android", 2014) agrupa en distintas categorías algunos ejemplos de las acciones por voz.

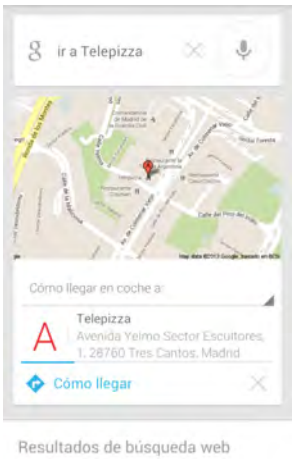
Categoría	Descripción	Ejemplo
Música	Permite al usuario escuchar música través del comando: "Escuchar" + Canción/Grupo de música. La aplicación de Búsqueda por voz abrirá un listado con las opciones que ofrece el dispositivo para escuchar la música solicitada (i.e. *YouTube* y otras aplicaciones de música como Pandora, *Play Music*, *TuneIn Radio*, Música, etc.)	- Escuchar: *Forever Young* (Figura 2.8.a). - Escuchar: Jarabe de Palo.
Mensajes	1. Envío de Mensajes: Permite enviar un mensaje de texto a otro usuario mediante el comando: "Enviar Mensaje"+ "A" + nombre del contacto + "Mensaje" + contenido del mensaje. 2. Envío de Correos: Permite enviar un correo a otro usuario mediante el comando: "Enviar Correo"+ "A" + nombre del contacto + "Asunto" + asunto del correo + "Mensaje" + mensaje de texto. 3. Notas: Se envía un correo al propio usuario con el recordatorio correspondiente. La dirección de correo es la de Gmail que tiene el usuario por defecto. Se envía el texto y el audio con el mensaje que ha dictado el usuario. La aplicación de búsqueda por voz mostrará una tarjeta con el nombre de la persona, su foto de contacto y el contenido del mensaje de correo o de texto que se quiere enviar.	- Enviar mensaje a Elena mensaje "Hola, ¿te apuntas esta noche a cenar?". - Enviar correo a Rosa Asunto "Trabajo" Mensaje "Reunión a las 11"(Figura 2.8.b). - Nota a mí: Comprar el pan.
Crear Eventos	Permite crear un evento en el calendario del dispositivo.	- Crear evento calendario cena el sábado a las 7 (Figura 2.8.d).
Mapas.	Permite solicitar el mapa de cualquier lugar.	- Mapa del Parque del Retiro, Madrid.

[4]las acciones de voz solo están disponibles en alemán, español, francés, inglés e italiano.

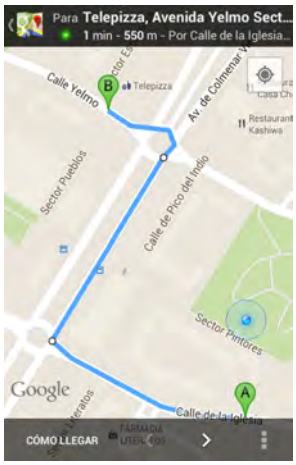
Recordatorios y alarmas	Permite establecer una alarma (i.e. “Establecer alarma” + “A las/dentro de” + hora) o un recordatorio (i.e. “Recordatorio” + Texto del mensaje).	- Recordar leer el correo en dos horas. - No olvidar cortar pelo. - Alarma mañana a las 7 (Figura 2.8.c). - Establecer alarma dentro de dos horas. - Crear evento calendario cena el sábado a las 7.
Llamadas	Permite realizar llamada a un contacto de la agenda del dispositivo. Si un contacto tiene varios teléfonos asociados, permite seleccionar el tipo de número de teléfono: Móvil, trabajo, o casa. Además permite llamar a cualquier establecimiento.	- Llamar a Telepizza Tres Cantos (Figura 2.8.e). - Llamar a Elena. - Llamar a Rosa trabajo.
Direcciones	El sistema de navegación de Android está integrado a la búsqueda por voz de Jelly Bean. Basta con decir al dispositivo donde se quiere ir y este le responderá con las indicaciones asociadas.	- Cómo llegar a Avenida de Fernández Ladreda 87, Segovia. - Llegar a la Plaza del Castillo, Pamplona. - Ir a Telepizza (Figura 2.7).
Abrir aplicación	Permite abrir cualquier aplicación	- Abrir Facebook (Figura 2.8.f). - Abrir Gmail.
Ir a una página web	Dirige al usuario a la página web que solicita.	- Ir a Google.com

Tabla 2.4: Posibilidades ofrecidas por las Acciones de voz en Android 4.1

La Figura 2.7 muestra un ejemplo de la aplicación de Google Maps de Android integrada a la aplicación de búsqueda por voz.



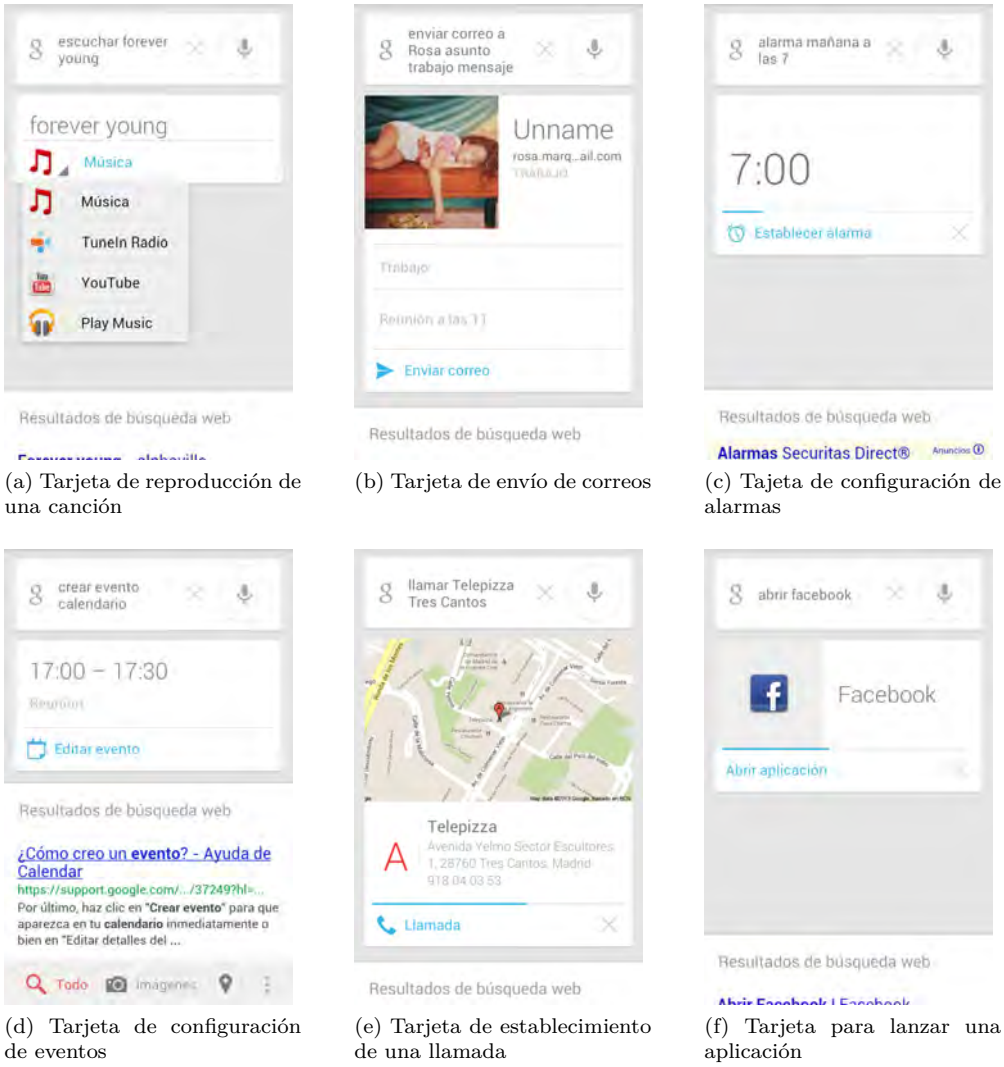
(a) Comando de voz : “Ir a” + Lugar destino



(b) Aplicación de Google Maps

Figura 2.7: Aplicación de Google Maps de Android integrada a la aplicación de búsqueda por voz

La Figura 2.8 muestra las capturas de pantalla asociadas a las respuestas de los ejemplos de la Tabla 2.4.



(a) Tarjeta de reproducción de una canción

(b) Tarjeta de envío de correos

(c) Tajeta de configuración de alarmas

(d) Tarjeta de configuración de eventos

(e) Tarjeta de establecimiento de una llamada

(f) Tarjeta para lanzar una aplicación

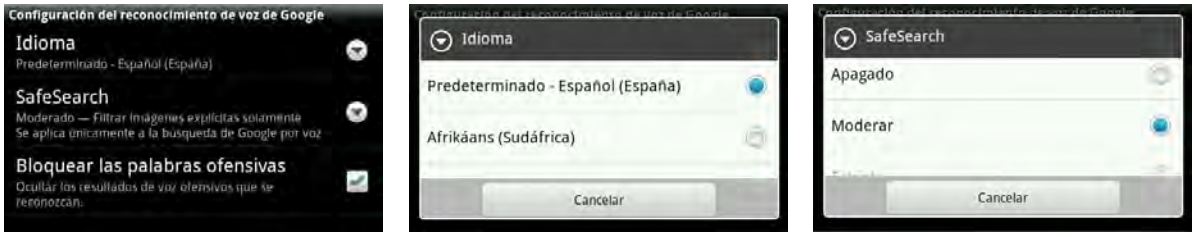
Figura 2.8: Respuestas a los comandos de acciones de voz para Android 4.1

2.2.1.4. Configuración de la búsqueda por voz de Google

Como se ha visto en las secciones anteriores, para que funcione tanto la entrada de voz por teclado como las Acciones de voz, es necesario que se encuentre instalada en el dispositivo la aplicación de búsqueda por voz de Google. En la mayoría de dispositivos dicha aplicación viene instalada por defecto. De no ser así se puede descargar en *Google Play* ("GooglePlay", 2014). Los ajustes de voz permiten controlar diferentes aspectos de la entrada de voz al hacer búsquedas por voz o al utilizar las acciones de voz.

Ajustes de la búsqueda por voz para dispositivos Android 2.2

Para entrar en el menú de configuración del reconocimiento de voz de Google el usuario debe dirigirse a Ajustes->Entrada y Salida de voz-> Reconocimiento de voz. La Figura 2.9.a muestra una captura de pantalla del menú de configuración del reconocimiento de voz de Google en un dispositivo con Android 2.2 en el que se observan las opciones de configuración.



(a) Menú de Configuración (b) Selección del idioma (c) Opción *SafeSearch*

Figura 2.9: Menú de Configuración del reconocimiento de voz de Google

Como se observa en dicha Figura, aparecen tres opciones:

- Idioma: Permite seleccionar el idioma de la entrada por voz. El idioma seleccionado para la entrada de la búsqueda por voz puede ser diferente del idioma utilizado en la interfaz del teléfono. La Figura 2.9.b muestra parte del menú que permite seleccionar el idioma para la entrada de la búsqueda por voz. Como se observa, el español aparece como el idioma predeterminado.

- *SafeSearch*: Los filtros *SafeSearch* de Google permiten cambiar la configuración del navegador para evitar que aparezca contenido explícito en los resultados de búsqueda. El filtro se basa principalmente en algoritmos que tienen en cuenta diversos factores, incluidos enlaces, palabras clave e imágenes. Al seleccionar esta opción dentro del menú de Configuración de reconocimiento de voz, el filtro *SafeSearch* se aplicará únicamente a la búsqueda de Google por voz. En la Figura 2.9.c aparece el menú que permite seleccionar entre tres opciones para el filtro SafeSearch:

  - Filtro estricto: Excluye imágenes y texto explícito de las páginas de resultados de la Búsqueda de Google.
  - Filtro moderado: Excluye únicamente imágenes explícitas de las páginas de resultados de la Búsqueda de Google. Esta es la opción predeterminada de SafeSearch.
  - Apagado: No filtra los resultados de las búsquedas por voz.

- Bloquear palabras ofensivas: Al seleccionar esta opción se bloquean los resultados de voz que contengan lenguaje ofensivo.

Ajustes de la búsqueda por voz para dispositivos Android 4.1 o superior

Para entrar en el menú de configuración del reconocimiento de voz de Google el usuario debe dirigirse a Ajustes->Idioma y Teclado->Búsqueda por voz. En dicho menú se puede configurar cualquiera de las opciones mostradas a continuación.

- Idioma: El idioma que se selecciona en esta sección para la entrada y la salida de la búsqueda por voz puede ser diferente del idioma utilizado en la interfaz del dispositivo. Google incorpora regularmente nuevos idiomas.

- Salida de voz: Para activar la salida de voz solo cuando el usuario tenga unos auriculares conectados al dispositivo, se selecciona la opción “Solo en manos libres”.

- Bloquear palabras ofensivas: Se selecciona esta opción para que se bloqueen los resultados de las búsquedas que contengan lenguaje ofensivo.

- Detección de palabras clave: Esta opción permite al usuario “Google” en la pantalla de *Google Now* en lugar de tener que tocar el icono de micrófono para iniciar una búsqueda por voz o una acción de voz.

- Reconocimiento sin conexión: Si el dispositivo no tiene conexión de red, esta opción permite seleccionar uno o varios idiomas más para el reconocimiento. El problema de esta funcionalidad es que no se encuentra disponible en todos los dispositivos y no existe ninguna documentación que haga referencia a esta funcionalidad y a que dispositivos se puede aplicar.

2.2.2. Integración del reconocimiento de voz en una aplicación para dispositivos móviles Android

El *SDK* (*Software Development Kit*) de Android permite integrar de forma sencilla y directa el reconocimiento de voz en las aplicaciones. Al tratarse Android de una plataforma abierta, las aplicaciones pueden utilizar cualquier servicio de reconocimiento voz que el dispositivo haya registrado para recibir un objeto de la clase android.speech.RecognizerIntent ("API", 2014). La aplicación de búsqueda por voz de Google (*Google Search*, en inglés), instalada por defecto en la mayoría de los dispositivos Android, responde a un RecognizerIntent mostrando una ventana con el texto "Habla ahora". Posteriormente transfiere el audio a los servidores de Google (los mismos que se utilizan cuando el usuario pulsa el botón del micrófono en el *widget* del buscador de Google o en el teclado con entrada de voz habilitada) (Gruenstein, 2010).

Asimismo, existen dos alternativas desarrolladas por terceros para la integración de reconocimiento de voz en una aplicación Android (TI, 2012):

- SDK de reconocimiento de voz y síntesis de texto a voz para Android desarrollado por la empresa iSpeech (`https://www.ispeech.org/developers/android`) para desarrolladores de aplicaciones: Permite integrar el reconocimiento de voz y la síntesis de texto a voz a cualquier aplicación Android mediante la nube de iSpeech. Dispone de 27 idiomas para el reconocimiento de voz y la síntesis de texto a voz y 15 idiomas para el dictado en la entrada por voz.

- CMUSphinx ("CMU Sphinx", 2013) : Kit de herramientas de código abierto desarrollado por *Carnegie Mellon University* para la integración de reconocimiento de voz en una aplicación Android. Permite integrar el reconocimiento de voz en modo *offline* a una aplicación, sin necesidad de disponer de acceso a Internet.

A pesar de existir varias alternativas para la integración del reconocimiento de voz en una aplicación Android, la opción más simple consiste en enviar un objeto de la clase android.speech.RecognizerIntent a la aplicación de búsqueda por voz de Google.

En la presente sección se resume las principales características del paquete android.speech del API de Android. Posteriormente, se realiza un estudio de la clase android.speech.RecognizerIntent al tratarse de la alternativa más simple y se describe como integrar el reconocimiento de voz en una aplicación de Android mediante la utilización de dicha clase.

2.2.2.1. Utilización del paquete android.speech

A través de las clases disponibles en el paquete android.speech, Android ofrece tanto la posibilidad de integrar en una aplicación servicios de reconocimiento de voz ya existentes como la de diseñar servicios de reconocimiento de voz completamente nuevos. La Tabla 2.5 ("API", 2014) resume las clases e interfaces del paquete android.speech. Para una explicación más detallada del paquete android.speech se puede consultar el API de Android.

Las aplicaciones no necesitan ningún permiso especial para acceder a los servicios de reconocimiento de voz de Android. Sin embargo, es necesario disponer una conexión de datos para que el reconocimiento de voz pueda funcionar ya que el reconocimiento de voz sin conexión esta únicamente disponible en cierto dispositivos.

Tal y como se ha explicado anteriormente, a la hora de integrar el reconocimiento de voz en una aplicación, la opción más simple consiste en llamar al *intent* correspondiente al reconocimiento de voz, utilizando para ello la clase android.speech.RecognizerIntent.

Interfaz	Definición
RecognitionListener	Recibe notificaciones del reconocedor de voz (SpeechRecognizer) cuando se ha dado algún evento relacionado con dicho reconocedor de voz (Ej. El usuario ha comenzado/terminado de hablar, los resultados del reconocimiento de voz están disponibles, etc.).

(a) Interfaces

Clase	Definición
RecognitionService	Clase que sirve de base para implementar un servicio de reconocimiento de voz. Se debe tener en cuenta que una clase podrá únicamente extender la clase RecognitionService si se quiere implementar un nuevo reconocedor de voz. Todos sus métodos son abstractos. Las notificaciones asociadas al servicio de reconocimiento de voz se enviarán a la clase RecognitionService.Callback.
RecognitionService. Callback	Clase que recibe las notificaciones del servicio de reconocimiento de voz y se las envía al usuario. Se deberá pasar una instancia de esta clase al método onStartListening(Intent, Callback) de la clase RecognitionService.
RecognizerIntent	Clase que define las constantes necesarias para integrar el reconocimiento de voz iniciado desde un *intent*.
RecognizerResultsIntent	Clase que define las constantes asociadas a *intents* que se encargan de mostrar los resultados del reconocimiento de voz. No es necesaria para la utilización estándar del reconocimiento de voz.
SpeechRecognizer	Clase que proporciona acceso al servicio de reconocimiento de voz, el cual permite tener acceso al reconocedor de voz. No se debe crear un objeto de esta clase directamente, sino que se deberá llamar al método estático createSpeechRecognizer(Context). Permite crear el RecognitionListener que recibirá todas las notificaciones en respuesta a los eventos asociados al reconocedor de voz. Para poder utilizar esta clase, la aplicación debe tener el permiso RECORD_AUDIO.

(b) Clases

Tabla 2.5: Clases e Interfaces del paquete android.speech

2.2.2.2. Utilización de la clase android.speech.RecognizerIntent

Como se explicó en el apartado anterior, la opción más simple para la integración del reconocimiento de voz en una aplicación consiste en lanzar el *intent* de tipo android.speech.RecognizerIntent. Este *intent* origina la activación del grabador de voz de Android solicitando al usuario que introduzca la entrada por voz. El fichero audio resultante se envía a los servidores de Google con el objetivo de ser procesado, por lo que es necesaria una conexión a Internet. Finalmente, los resultados del reconocimiento de voz son devueltos a la actividad que lanzó dicho *intent*.

La Tabla 2.6 ("API", 2014) resume las constantes de la clase android.speech.RecognizerIntent que se utilizan en la integración del reconocimiento de voz en una aplicación. Para una explicación más detallada de las constantes de la clase android.speech.RecognizerIntent se puede consultar el API de Android.

Tipo	Nombre de la Constante	Definición
String	ACTION_GET_LANGUAGE_ DETAILS	Invoca un *intent* de tipo *broadcast* que se envía a un objeto de la clase BroadcastReceiver con el fin de solicitar la lista de los idiomas disponibles del reconocedor de voz. El BroadcastReceiver está especificado en los metadatos DETAILS_META_DATA de la actividad que satisface ACTION_WEB_SEARCH. Cuando el *intent* se invoca

Tipo	Nombre de la Constante	Definición
		mediante el método sendOrderedBroadcast(Intent, String, BroadcastReceiver, android.os.Handler, int, String, android.os.Bundle), se devuelve el resultado de tipo Bundle al método onReceive() del BroadcastReceiver proporcionado. Dicho resultado contendrá el valor EXTRA_LANGUAGE_PREFERENCE o EXTRA_SUPPORTED_LANGUAGES. Es recomendable que las actividades que implementen ACTION_WEB_SEARCH proporcionen esta información pero no es obligatorio.
String	ACTION_RECOGNIZE_SPEECH	*Intent* que inicia la actividad encargada de solicitar al usuario que realice su consulta vía voz y de enviarla al reconocedor. Los resultados se pueden devolver a través del método onActivityResult(int, int, Intent) si se lanza el *intent* mediante el método startActivityForResult(Intent, int), o se pueden enviar a través de un PendingIntent si se proporciona.
String	ACTION_VOICE_SEARCH_ HANDS_FREE	*Intent* que inicia la actividad encargada de solicitar al usuario que realice su consulta vía voz y de enviarla al reconocedor. No será necesario que el usuario visualice la pantalla para obtener la respuesta ya que o bien se sintetiza el resultado si se trata de una búsqueda web, o se realiza directamente la acción correspondiente a la petición del usuario.
String	ACTION_WEB_SEARCH	*Intent* que inicia la actividad encargada de solicitar al usuario que realice su consulta vía voz y de enviarla al reconocedor. Se muestra el resultado de la búsqueda por pantalla o se realiza directamente la acción correspondiente a la petición del usuario.
String	DETAILS_META_DATA	Nombre de los metadatos mediante el cual una actividad que implemente ACTION_WEB_SEARCH puede utilizar para revelear el nombre de la clase BroadcastReceiver, encargada de responder a cualquier solicitud de información por parte de los *intent* de tipo broadcast existentes en la clase RecognizerIntent.
String	EXTRA_CALLING_PACKAGE	Se utiliza como clave extra en el *intent* de búsqueda por voz. Se añade a través de la llamada al método putExtra(String name, String value). El diálogo de búsqueda utiliza dicho extra para establecer un paquete que identifique la aplicación de búsqueda por voz. La especificación de dicho extra es opcional.
String	EXTRA_CONFIDENCE_SCORES	*Array* que contiene valores de tipo *float* que representan la fiabilidad de cada resultado devuelto por la actividad correspondiente al *intent* ACTION_RECOGNIZE_SPEECH. La especificación de dicho extra es opcional.
String	EXTRA_LANGUAJE	Indica al reconocedor que realice el reconocimiento de voz en el idioma especificado por un código de idioma IETF, tal como define el estándar BCP 47 (Ej. “en-US”). La especificación de dicho extra es opcional.
String	EXTRA_LANGUAGE_MODEL	Informa al reconocedor sobre el modelo del lenguaje utilizado por el ACTION_RECOGNIZE_SPEECH:

Tipo	Nombre de la Constante	Definición
		LANGUAGE_MODEL_WEB_SEARCH para frases de búsqueda o LANGUAGE_MODEL_FREE_FORM para dictado. La especificación de dicho extra es obligatoria.
String	EXTRA_LANGUAGE_ PREFERENCE	Es la clave que identifica el contenido extra del resultado de tipo Bundle devuelto por el *intent* ACTION_GET_LANGUAJE_DETAILS a través del BroadcastReceiver. Se trata de una cadena que identifica el idioma que ha especificado el usuario (Ej. "en-US").
String	EXTRA_MAX_RESULTS	Límite opcional del máximo número de resultados del reconocimiento de voz. Es opcional. Si se omite será el reconocedor quien elija el máximo de resultados a devolver.
String	EXTRA_ONLY_RETURN_ LANGUAGE_PREFERENCE	Se especifica este booleano como extra al *intent* ACTION_GET_LANGUAJE_DETAILS cuando se quiere que el único resultado devuelto sea el resultado EXTRA_LANGUAJE_PREFERENCE.
String	EXTRA_ORIGIN	Valor opcional que especifica la URL de la página en la que se solicito el habla.
String	EXTRA_PARTIAL_RESULTS	Valor booleano que indica si se deben devolver resultados parciales a medida que el usuario está hablando (puesto a *false* por defecto). Es opcional.
String	EXTRA_PROMPT	Texto opcional que se mostrará al usuario cuando se deba comenzar a hablar.
String	EXTRA_RESULTS	Los resultados del reconocimiento de voz de tipo ArrayList<String> cuándo se ha realizado el *intent* ACTION_RECOGNIZE_SPEECH. Normalmente dicha lista se ordena en orden decreciente en cuanto a la fiabilidad del reconocedor de voz dado por EXTRA_CONFIDENCE_SCORES. Únicamente se devuelven dichos resultados cuando la actividad devuelve el código RESULT_OK.
String	EXTRA_RESULTS_PENDING_ INTENT	Extra que sirve para proporcionar un PendingIntent devuelto por la llamada al método getActivities(Context, int, Intent,int, Bundle) de forma que los resultados del reconocimiento se añadan al parámetro Bundle. Posteriormente el PendingIntent se envía al destino.
String	EXTRA_RESULTS_PENDING_ INTENT_BUNDLE	Si se utiliza EXTRA_RESULTS_PENDING_INTENT para proporcionar un PendingIntent, se puede utilizar EXTRA_RESULTS_PENDING_INTENT_BUNDLE para añadir extras adicionales al PendingIntent que se envía al destino además de los resultados de reconocimiento de voz.
String	EXTRA_SECURE	Valor opcional booleano que indica si una búsqueda por voz en manos libres se efectuó en el modo seguro. (Ej. Pantalla del dispositivo bloqueada).
String	EXTRA_SPEECH_INPUT_ COMPLETE_SILENCE_ LENGTH_MILLIS	Extra que especifica el tiempo que debe pasar después de escuchar la voz para poder considerarse que se ha terminado de decir la petición.
String	EXTRA_SPEECH_INPUT_ LENGTH_MILLIS	Extra que especifica la longitud mínima de una elocución. No se parará la grabación antes de este periodo de tiempo.

Tipo	Nombre de la Constante	Definición
String	EXTRA_SPEECH_INPUT_ POSSIBLY_COMPLETE_ SILENCE_LENGTH_MILLIS	Extra que especifica el tiempo que debe pasar después de escuchar la voz para poder considerarse que se ha terminado posiblemente de decir la petición.
String	EXTRA_SUPPORTED_ LANGUAJES	La clave que identifica el contenido extra del resultado de tipo Bundle devuelto por el *intent* ACTION_GET_LANGUAJE_DETAILS a través del BroadcastReceiver. Es un ArrayList<String> y contiene los idiomas disponibles en la implementación actual del reconocimiento.
String	EXTRA_WEB_SEARCH_ONLY	Booleano utilizado junto al *intent* ACTION_WEB_SEARCH que indica si solamente deben mostrarse las respuestas a las búsquedas web que respondan a la petición del usuario. Por defecto puesto a *false* para que también se puedan realizar otro tipo de acciones como consecuencia de lo que solicite el usuario. Es opcional.
String	LANGUAGE_MODEL_FREE_ FORM	Establece el modelo de lenguaje FREE FORM para dictado. Se usa junto a la clave EXTRA_LANGUAJE_MODEL.
String	LANGUAGE_MODEL_WEB_ SEARCH	Establece el modelo de lenguaje WEB_SEARCH para búsquedas de términos en la web. Se usa junto a la clave EXTRA_LANGUAJE_MODEL.
int	RESULT_AUDIO_ERROR	Código de respuesta cuando se da un error de audio.
int	RESULT_CLIENT_ERROR	Código de respuesta cuando se da un error genérico del cliente.
int	RESULT_NETWORK_ERROR	Código de respuesta cuando se da un error en la red.
int	RESULT_NO_MATCH	Código de respuesta cuando no se ha encontrado ninguna coincidencia con las palabras dichas.
int	RESULT_SERVER_ERROR	Código de respuesta cuando un servidor de reconocimiento de voz devuelve un error.

Tabla 2.6: Constantes de la clase RecognizerIntent

De la Tabla 2.6 cabe destacar dos aspectos importantes que deben tenerse en cuenta a la hora de integrar el reconocimiento de voz en una aplicación (TI, 2012).

- Modelo de lenguaje: Con la intención de que la entrada por voz sea lo más precisa posible, los servicios de reconocimiento de voz disponibles en el SDK de Android ofrecen al desarrollador dos modelos de lenguaje entre los cuales debe elegir atendiendo al tipo de frases que se van a introducir: *free form* y *web search*. El modelo *free form* está pensado para mejorar la precisión de dictado en la entrada por voz a través del teclado en situaciones en las que se quiere escribir un mensaje SMS, un email, etc. Por otro lado, el modelo *web search* está pensado para búsquedas por voz, en las que se emplean frases cortas de búsqueda (Ej. "Temperatura Madrid"). Además de los servicios de voz existentes en el SDK de Android que permiten cualquiera de estos dos modelos, Android permite desarrollar servicios de voz propios utilizando las clases disponibles en el paquete android.speech.

- Idiomas disponibles en los servidores Google: Los servidores de Google disponen de multitud de idiomas para la entrada de voz que van incorporándose regularmente. Se puede utilizar el *intent* ACTION_GET_LANGUAJE_DETAILS para obtener el listado de idiomas disponibles. El modelo *web search* está disponible para todos los idiomas, mientras que el *free form* puede no estar optimizado para todos los idiomas.

La clase android.speech.RecognizerIntent consta de un único método denominado IntentgetVoiceDetailsIntent(Context context) descrito en la Tabla 2.7 (“API”, 2014) .

Método	public static final IntentgetVoiceDetailsIntent(Context context)
Definición	Devuelve un *intent* de tipo broadcast que se lanza mediante el método sendOrderedBroadcast(Intent,String, BroadcastReceiver, android.os.Handler,int, String, Bundle) con el objetivo de recibir información del paquete que implementa la búsqueda por voz. Está basado en el valor especificado en los metadatos DETAILS_META_DATA de la actividad encargada de la búsqueda por voz. Si dicho valor no está especificado devuelve *null*. También devuelve null si no se no se elige una actividad que responda por defecto al ACTION_WEB_SEARCH.

Tabla 2.7: Método IntentgetVoiceDetailsIntent(Context context)

2.2.2.3. Integración del reconocimiento de voz en una aplicación para dispositivos móviles Android mediante la clase android.speech.RecognizerIntent

Tras haber descrito en el apartado anterior las constantes de la clase android.speech.RecognizerIntent, se explican los pasos a seguir a la hora de integrar el reconocimiento de voz en una aplicación Android.

En primer lugar, es necesario dar a la aplicación los permisos necesarios para que soporte el reconocimiento de voz. Dado que para que funcione el reconocimiento de voz es necesario tener acceso a Internet, se dará dicho permiso en el archivo **AndroidManifest.xml** a través de la línea de código mostrada en la Figura 2.10.

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Figura 2.10: Permiso de acceso a Internet en el archivo AndroidManifest.xml

Una vez realizado este paso previo, se comienza a implementar el reconocimiento de voz. Toda aplicación que integre reconocimiento de voz debe realizar tres funciones principales descritas en los subsiguientes párrafos (Soto, 2014).

Verificar que existe la actividad para reconocer voz

Con el objetivo de comprobar si el reconocimiento de voz está disponible, se incluye el código mostrado en la Figura 2.11 (Soto, 2014).

```
private boolean ExisteActividadReconocerVoz(){
    PackageManager pm = getPackageManager();
    List<ResolveInfo> activities = pm.queryIntentActivities(
            new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH), 0);
    if (activities.size() != 0) {
        return true;
    } else {
        return false;
    }
}
```

Figura 2.11: Código desarrollado para verificar que existe la actividad para reconocer voz

La clase PackageManager ("API", 2014) obtiene información de los paquetes que están instalados en el dispositivo. Para obtener una instancia de dicha clase se invoca al método getPackageManager(). Dicha instancia invoca al método queryIntentActivities(Intent intent, int flags), que devuelve la lista de actividades asociadas al *intent* que se le pasa como primer parámetro. Para el caso de reconocimiento de voz, el primer parámetro es el *intent* RecognizerIntent.ACTION_RECOGNIZE_SPEECH, por lo que la llamada a dicho método devuelve el número de actividades capaces de realizar un reconocimiento de voz. Si el tamaño de dicha lista es distinto de cero, implica que el servicio de reconocimiento de voz está disponible en el dispositivo y se procede a invocar a la actividad de reconocimiento de voz.

Invocar a la actividad de reconocimiento de voz

La implementación del método que invoca a la actividad de reconocimiento de voz se muestra en la Figura 2.12 (Soto, 2014).

```
private void iniciarActividadReconocimientoDeVoz() {
    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
                    RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
    intent.putExtra(RecognizerIntent.EXTRA_PARTIAL_RESULTS, false);
    intent.putExtra(RecognizerIntent.EXTRA_MAX_RESULTS, 2);
    intent.putExtra(RecognizerIntent.EXTRA_PROMPT,
                    "Demostración del Reconocimiento de voz ;)");
    startActivityForResult(intent, VOICE_RECOGNITION_REQUEST_CODE);
}
```

Figura 2.12: Código desarrollado para invocar a la actividad de reconocimiento de voz

En primer lugar, se crea un objeto de la clase Intent ("API", 2014) cuya acción es del tipo RecognizerIntent.ACTION_RECOGNIZE_SPEECH. Dicha actividad es la que devuelve los resultados del reconocimiento.

Posteriormente, se configuran los parámetros de la actividad de reconocimiento de voz a través de la llamada del objeto de la clase Intent a putExtra(String name, String value) que especifica un par <clave,valor>. Estos parámetros pueden ser cualquiera de los estudiados en la Tabla 2.6, siendo únicamente obligatorio especificar el modelo de lenguaje utilizado (WEB_SEARCH para frases de búsqueda o FREE_FORM para dictado) a través de la clave EXTRA_LANGUAJE_MODEL. En el ejemplo de la Figura 2.12 se especifica el modelo de lenguaje FREE_FORM. Además, se especifica que no se devuelvan resultados parciales a través de la clave EXTRA_PARTIAL_RESULTS, se establece que se devuelvan como máximo dos resultados a través de la clave EXTRA_MAX_RESULTS, y se especifica el mensaje que se mostrará al usuario cuando se solicite que hable a través de la clave EXTRA_PROMPT.

Finalmente, se llama al método startActivityForResult(Intent intent, int requestCode), que se encarga de invocar a la actividad de reconocimiento de manera que devuelva los resultados del reconocimiento a través del método onActivityResult(int requestCode, int resultCode, Intent data). La constante VOICE_RECOGNITION_REQUEST_CODE se utiliza para identificar a la actividad que invoca a dicho *intent* y se declara al principio de la actividad.

Procesar los resultados

En la Figura 2.13 (Soto, 2014) se sobrescribe el método onActivityResult(int requestCode, int resultCode, Intent data) de la clase Activity ("API", 2014) que se encarga de recibir los resultados del reconocimiento. Dicho método se invoca cuando una actividad secundaria lanzada desde la actividad principal devuelve un resultado. Por ejemplo, después de reconocer lo que el usuario ha dicho, devolverá un resultado con las posibles frases que ha reconocido el servicio.

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == VOICE_RECOGNITION_REQUEST_CODE && resultCode == RESULT_OK) {

            ArrayList<String> matches = data.getStringArrayListExtra(
                    RecognizerIntent.EXTRA_RESULTS);
            super.onActivityResult(requestCode, resultCode, data);
    }
}
```

Figura 2.13: Código desarrollado para procesar los resultados del reconocimiento de voz

En primer lugar, se verifica la variable *requestCode* que identifica a la actividad que solicitó el reconocimiento de voz. También se verifica que el resultado devuelto por la actividad secundaria que realiza el reconocimiento de voz haya sido satisfactorio a través de la variable *resultCode*. Si la variable *requestCode* coincide con el valor VOICE_RECOGNITION_REQUEST_CODE que se pasó como parámetro al método startActivityForResult(Intent intent, int requestCode) y el código de respuesta *resultCode* tiene el valor RESULT_OK se pueden procesar los resultados devueltos por la actividad de reconocimiento de voz. Una vez realizadas las comprobaciones, se almacenan los resultados del reconocimiento en un ArrayList de *strings*. A partir de aquí, cómo se utilicen los resultados ya es cosa del desarrollador de la aplicación.

2.2.2.4. Integración de la entrada de voz por teclado en dispositivos Android 4.0 o superior

Tal y como se ha explicado en secciones anteriores, con la llegada de Android 4.0 se ha incorporado una nueva característica a la entrada de voz por teclado: los resultados del reconocimiento se transcriben en el campo de texto a medida que el usuario habla, sin esperar que éste haya terminado.

Según Zanolin (Zanolin, 2011), para integrar dicha característica a un método de entrada (*IME - Input method*) de forma simple se deben seguir los siguientes pasos.

1. Descargar la librería desarrollada ("Voice IME", 2014) para integrar dicha característica a la entrada de voz y añadirla al IME APK.

2. Crear una instancia de la clase VoiceRecognitionTrigger ("Voice IME", 2014) perteneciente a la librería descargada dentro del método onCreate() de la clase InputMethodService ("API", 2014) tal y como se observa en la Figura 2.14.

```
public void onCreate() {
    super.onCreate();
    ...
    mVoiceRecognitionTrigger = new VoiceRecognitionTrigger(this);
}
```

Figura 2.14: Código desarrollado para crear una instancia de la clase VoiceRecognitionTrigger

3. Añadir el icono del micrófono a la interfaz del IME: Para ello, se debe instanciar un objeto View.OnClickListener ("API", 2014) para lanzar el reconocimiento de voz. El botón del micrófono debe mostrarse únicamente si el reconocimiento de voz está disponible en el dispositivo. Para realizar dicha comprobación, la instancia VoiceRecognitionTrigger invoca al método isInstalled(). Se

puede especificar en qué idioma se desea realizar el reconocimiento de voz mediante un parámetro del método startVoiceRecognition(java.lang.String language) de la clase VoiceRecognitionTrigger. La Figura 2.15 muestra la implementación del presente paso.

```
public View onCreateInputView() {
  LayoutInflater inflater = (LayoutInflater) getSystemService(
      Service.LAYOUT_INFLATER_SERVICE);
  mView = inflater.inflate(R.layout.ime, null);
  ...
  mButton = (ImageButton) mView.findViewById(R.id.mic_button);
  if (mVoiceRecognitionTrigger.isInstalled()) {
    mButton.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View v) {
        mVoiceRecognitionTrigger.startVoiceRecognition();
      }
    });
    mButton.setVisibility(View.VISIBLE);
  } else {
    mButton.setVisibility(View.GONE);
  }
  return mView;
}
```

Figura 2.15: Código desarrollado para añadir el icóno del micrófono a la interfaz del IME

4. Notificar a la instancia de la clase VoiceRecognitionTrigger en el instante que se arranque el IME tal y como se observa en la Figura 2.16, para que pueda introducir en la vista los resultados del reconocimiento de voz pendientes.

```
@Override
public void onStartInputView(EditorInfo info, boolean restarting) {
  super.onStartInputView(info, restarting);
  if (mVoiceRecognitionTrigger != null) {
    mVoiceRecognitionTrigger.onStartInputView();
  }
}
```

Figura 2.16: Código desarrollado para notificar del comienzo del IME

5. Modificar el fichero AndroidManifest.xml: Para realizar el reconocimiento de voz, se utiliza un servicio (ServiceHelper) y una actividad (ActivityHelper) ("Voice IME", 2014) pertenecientes a la librería descargada, por lo que se deben añadir al fichero AndroidManifest.xml tal y como se observa en la Figura 2.17.

```
<manifest ... >
  <application ...>
    ...
    <service android:name="com.google.android.voiceime.ServiceHelper" />
    <activity
        android:name="com.google.android.voiceime.ActivityHelper"
        android:theme="@android:style/Theme.Translucent.NoTitleBar"
        android:excludeFromRecents="true"
        android:windowSoftInputMode="stateAlwaysHidden"
        android:finishOnTaskLaunch="true"
        android:configChanges="keyboard|keyboardHidden|navigation
                               |orientation"/>
  </application>
</manifest>
```

Figura 2.17: Modificación del fichero AndroidManifest.xml para implementar la transcripción por voz

6. Actualizar el icono del micrófono dinámicamente (opcional): El reconocimiento de voz necesita en la mayoría de dispositivos acceso a Internet para funcionar, por lo que si no se encuentra Internet, el IME debe notificar al usuario de que el reconocimiento de voz se encuentra deshabilitado. Para ello, se debe registrar un objeto VoiceRecognitionTrigger.Listener ("Voice IME", 2014) y habilitar/deshabilitar el micrófono dependiendo del acceso a Internet tal y como se observa en la Figura 2.18. El objeto VoiceRecognitionTrigger.Listener se registra dentro del método onCreate() y deshabilita dentro del método onDestroy(), ambos de la clase InputMethodService ("API", 2014).

```
public void onCreate() {
  super.onCreate();
  ...
  mVoiceRecognitionTrigger = new VoiceRecognitionTrigger(this);
  mVoiceRecognitionTrigger.register(new VoiceRecognitionTrigger.Listener() {
    @Override
    public void onVoiceImeEnabledStatusChange() {
      updateVoiceImeStatus();
    }
  });
}
...
@Override
public void onDestroy() {
  ...
  if (mVoiceRecognitionTrigger != null) {
    mVoiceRecognitionTrigger.unregister(this);
  }
  super.onDestroy();
}
private void updateVoiceImeStatus() {
  if (mVoiceRecognitionTrigger.isInstalled()) {
    mButton.setVisibility(View.VISIBLE);
    if (mVoiceRecognitionTrigger.isEnabled()) {
      mButton.setEnabled(true);
    } else {
      mButton.setEnabled(false);
    }
  } else {
    mButton.setVisibility(View.GONE);
  }
  mView.invalidate();
}
```

Figura 2.18: Código desarrollado para actualizar dinámicamente el icono del micrófono

Finalmente, se añade el permiso ACCESS_NETWORK_STATE en el fichero AndroidManifest.xml tal y como se observa en la Figura 2.19.

```
<manifest ... >
  ...
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
  ...
</manifest>
```

Figura 2.19: Permiso ACCESS_NETWORK_STATE en el fichero AndroidManifest.xml

Cabe destacar que la presente modificación del IME funciona para dispositivos Android 2.2 y superior. Para Android 4.0 y superior, los usuarios podrán leer en la vista lo que van diciendo al reconocedor, sin necesidad de haber terminado de hablar. Para versiones anteriores, el comportamiento del reconocimiento de voz será el estándar.

2.3. Síntesis de Texto a Voz en Android

2.3.1. Introducción a la síntesis de voz en Android

El sistema operativo Android 1.6 o superior incorpora en la mayoría de sus dispositivos un motor de síntesis de voz denominado Pico TTS desarrollado por la compañía SVOX y Google, el cual permite integrar la síntesis de voz a cualquier aplicación. Mediante el motor de síntesis de voz, las aplicaciones son capaces de producir voz a partir de cualquier cadena de texto, permitiendo al usuario interactuar con la aplicación sin la necesidad de leer la pantalla del dispositivo (Hashimi et al., 2010).

Figura 2.20: Menú de ajustes de la síntesis de voz en dispositivos Android 1.6 o superior

Para poder escuchar un ejemplo del sintetizador de voz Pico TTS, una vez situados en el menú de ajustes del dispositivo se debe seleccionar: Entrada y salida de voz->Ajustes de síntesis de voz. Se llega al menú de la Figura 2.20, el cual consta de las opciones:

- **Escuchar un ejemplo**: Permite escuchar un ejemplo del motor de síntesis que se encuentre seleccionado en la opción **Motor predeterminado** y en el idioma que se encuentre seleccionado en la opción **Idioma**.
- **Utilizar siempre mi configuración**: Permite anular la configuración de cualquier aplicación que utilice el motor TTS. Por ello es preferible dejar esta opción sin seleccionar si se desea que la aplicación tenga el comportamiento esperado.

Bajo la etiqueta de ajustes predeterminados aparecen las opciones:

- **Motor predeterminado**: Establece el motor de síntesis de voz utilizado para el texto hablado. En los dispositivos Android 1.6 o superior viene incorporado el motor Pico TTS por defecto por lo que solo aparece esta opción si no se ha instalado ningún otro motor. No obstante, en las últimas versiones de Android, el motor de síntesis que viene incorporado por defecto es Google TTS que ofrece unas voces de calidad muy similar a las del motor Pico TTS.

- **Instalar archivos de datos de voz**: Permite instalar los archivos de voz necesarios para la síntesis de voz. Es necesario que esté instalado el paquete *SpeechSynthesis Data Installer* ("GooglePlay", 2014), que contiene los ficheros necesarios para la síntesis de texto a voz en los idiomas inglés EEUU/UK, francés, italiano, alemán y español. Si únicamente se quiere instalar el idioma español, basta con instalar *SpeechSynthesis SPA-ESP Voice* ("GooglePlay", 2014).

- **Velocidad de voz**: Permite seleccionar la velocidad a la que se lee el texto entre los valores **Muy lenta** y **Muy rápida**.

- **Idioma**: Establece la voz del idioma específico para el texto hablado. Pico TTS incorpora los idiomas: Inglés EEUU/UK, francés, italiano, alemán y español.

Bajo la etiqueta de motores aparecen los motores de síntesis que se encuentran instalados en el dispositivo. Por el momento, aparece únicamente el motor Pico TTS. En la Sección 2.3.3 se estudian otras posibilidades para el motor de síntesis, que una vez instalados, aparecen también en dicho menú.

Se va a profundizar en lo que sucede realmente cuando se entra en el menú de ajustes de síntesis de voz mostrado en la Figura 2.20. La actividad correspondiente a dicho menú inicializa el motor de síntesis configurándolo a la velocidad e idioma que hayan sido establecidos. Al seleccionar la opción **Escuchar un ejemplo**, la actividad envía un texto al motor de síntesis de voz Pico TTS, y este se encarga de reproducirlo en la salida de audio. Además de aportar simplicidad en el desarrollo de aplicaciones que incorporen TTS, el motor Pico TTS apenas consume memoria y espacio en disco, convirtiéndose en una mejora importante para los dispositivos (Hashimi et al., 2010).

El problema que tiene el motor Pico TTS es que, como se verá en la Sección 2.3.3, en comparación con otros motores de síntesis disponibles en Android, la calidad de la voz es regular, ya que resulta algo monótona y robótica. Si se pretende leer un bloque extenso de texto no es el motor de síntesis más adecuado.

Independientemente del número motores de síntesis de voz que se encuentren instalados en el dispositivo, únicamente habrá un motor de síntesis de voz activo que será compartido por todas las actividades que quieran hacer uso de él. Por ello, una actividad no sabrá el instante en el que se escuchará el texto que se envié al motor. Para tratar este problema, Android proporciona un mecanismo que notifica a la actividad sobre el estado del texto que se envió al motor (Hashimi et al., 2010). Dicho mecanismo se estudia en la Sección 2.3.2.4.

2.3.2. Integración de la síntesis de texto a voz en una aplicación para dispositivos móviles Android

La lógica bajo el motor de síntesis de voz es complicada, sin embargo, con Android 1.6 o superior es posible desarrollar aplicaciones que se encarguen únicamente de enviar el texto al motor de síntesis de voz para que este lo reproduzca, sin la necesidad de conocer su lógica interna (Hashimi et al., 2010). Para ello, se debe utilizar el paquete android.speech.tts de la API de Android, en particular la clase android.speech.tts.TextToSpeech que permite introducir en una cola el texto que se quiere reproducir.

Otra alternativa para integrar la síntesis de voz en una aplicación Android consiste en utilizar el SDK desarrollado por la empresa iSpeech. Sin embargo, la opción más simple es la de enviar el texto al motor de síntesis mediante la utilización de la clase android.speech.tts.TextToSpeech del API de Android por lo que se implementará la síntesis de texto a voz mediante la utilización de dicha clase.

A continuación, se resume las clases e interfaces que componen el paquete android.speech.tts y los aspectos a tener en cuenta a la hora de integrar la síntesis de texto a voz en una aplicación Android.

2.3.2.1. Utilización del paquete android.speech.tts

La Tabla 2.8 ("API", 2014) describe las clases e interfaces del paquete android.speech.tts . Para una explicación más detallada del paquete android.speech.tts se puede consultar el API de Android.

Interfaz	Definición
SynthesisCallback	Define el método que recibe el resultado de la síntesis de texto a voz realizada por el motor de síntesis.
TextToSpeech.OnInitListener	Define el método invocado cuando finaliza la inicialización del motor de síntesis.
TextToSpeech. OnUtteranceCompletedListener	Define la llamada invocada cuando el motor de síntesis completa la síntesis de una cadena de texto.

(a) Interfaces

Clase	Definición
SynthesisRequest	Contiene los datos que requiere el motor para realizar la síntesis de texto a voz: El idioma, el país y variante del texto a ser sintetizado, y la velocidad y tono de la voz.
TextToSpeech	Se encarga de la síntesis de texto a voz. El resultado de la síntesis será reproducido o guardado en un fichero de audio. Una instancia de la clase TextToSpeech puede únicamente sintetizar un texto cuando se haya completado su inicialización. Se debe implementar la interfaz TextToSpeech.OnInitListener para notificar a la aplicación una vez que la inicialización haya finalizado.
TextToSpeech.Engine	Contiene las constantes y nombres de parámetros que se utilizan para el control de la síntesis de texto a voz. Estos describen: ■ *Intents* que comprueban la disponibilidad de los datos y extras del motor TTS y solicitan la instalación de los mismos si es necesario. ■ Claves de la estructura HashMap que son pasados como parámetro del método speak(String, int, HashMap). ■ Lista de características que el motor puede utilizar. Estos valores se especifican en los métodos speak(String, int, HashMap) y synthesizeToFile(String, HashMap, String). Se puede solicitar las características que tiene el motor de síntesis a través de la llamada al método getFeatures(java.util.Locale) de la clase TextToSpeech.
TextToSpeech.EngineInfo	Solicita la información del motor de síntesis de texto a voz instalado en el dispositivo.
TextToSpeechService	Clase abstracta que permite implementar los siguientes métodos del motor de síntesis: ■ onIsLanguageAvailable(String, String, String): Comprueba la disponibilidad de un idioma del motor de síntesis. ■ onLoadLanguage(String, String, String): Indica al motor de síntesis la necesidad de cargar un determinado idioma para la síntesis de voz. ■ onGetLanguage(): Devuelve el idioma, el país y variante que está siendo utilizado por el motor de síntesis. ■ onSynthesizeText(SynthesisRequest, SynthesisCallback): Implementa la síntesis de texto a voz según los parámetros de síntesis especificados en el primer parámetro de la clase SynthesisRequest. Una vez realizada la síntesis de texto a voz, se envía el resultado a través de un método definido en la clase a la que hace referencia el segundo parámetro de la clase SynthesisCallback. ■ onStop(): Indica al motor de síntesis que debe de parar cualquier síntesis de texto a voz que se esté efectuando.
UtteranceProgressListener	Controla los eventos relacionados con el progreso de una cadena de texto dentro de la cola que contiene todas las cadenas a ser sintetizadas.

(b) Clases

Tabla 2.8: Clases e Interfaces del paquete android.speech.tts

2.3.2.2. Solicitud de los recursos necesarios para llevar a cabo la síntesis de texto a voz e inicialización del motor de síntesis

La voz y el diccionario son recursos específicos del idioma que se haya seleccionado y se deben cargar antes de que el motor de síntesis pueda sintetizar un texto. Algunos dispositivos tienen almacenamiento limitado y no traen consigo los archivos relacionados con estos recursos. No obstante, el API de Android permite consultar a la plataforma sobre la disponibilidad de los archivos de datos de voz e iniciar su descarga e instalación si es necesario (Trivi, 2009).

El primer paso en cualquier aplicación que integre TTS es comprobar la existencia de los recursos TTS. Para ello se crea el *intent* TextToSpeech.Engine.ACTION_CHECK_TTS_DATA ("API", 2014) tal y como muestra la Figura 2.21 (Trivi, 2009). La posterior llamada al método startActivityForResult(Intent intent, int requestCode) de la clase Activity, inicia una actividad que devuelve un código de respuesta a la aplicación a través del método onActivityResult(int requestCode, int resultCode, Intent data), también de la clase Activity. La constante MY_DATA_CHECK_CODE se utiliza en el método startActivityForResult(Intent intent, int requestCode) para identificar a la actividad que lo llama. La declaración de dicha constante se realiza al principio de la actividad.

```
Intent intent = new Intent(Engine.ACTION_CHECK_TTS_DATA);
startActivityForResult(intent, MY_DATA_CHECK_CODE );
```

Figura 2.21: Código desarrollado para solicitar la disponibilidad de recursos TTS

```
protected void onActivityResult(
        int requestCode, int resultCode, Intent data) {
    if (requestCode == MY_DATA_CHECK_CODE) {
        if (resultCode == TextToSpeech.Engine.CHECK_VOICE_DATA_PASS) {
            // Existen recursos TTS, por lo que se crea la instancia TTS
            mTts = new TextToSpeech(this, this);
        } else {
            // No existen recursos TTS por lo que se solicita la instalación
            Intent installIntent = new Intent();
            installIntent.setAction(
                    TextToSpeech.Engine.ACTION_INSTALL_TTS_DATA);
                    startActivity(installIntent);
        }
    }


}
```

Figura 2.22: Código desarrollado para descargar e instalar los ficheros de voz y crear un objeto de la clase TextToSpeech

La Figura 2.22 (Trivi, 2009) muestra el método onActivityResult(int requestCode, int resultCode, Intent data). En primer lugar, se comprueba que el código *requestCode* coincida con el código MY_DATA_CHECK_CODE y de ser así se procede a comprobar el código de respuesta *resultcode* devuelto por la actividad secundaria encargada de comprobar la disponibilidad de los archivos de voz. Si los archivos de voz se encuentran disponibles en el dispositivo, dicho código de respuesta almacena el valor TextToSpeech.Engine.CHECK_VOICE_DATA_PASS ("API", 2014) , que indica que el dispositivo está preparado para realizar la síntesis de texto a voz una vez se haya creado el objeto android.speech.tts.TextToSpeech. Cualquier otro código de respuesta significa que no están disponibles los ficheros de voz por lo que se informa al usuario sobre la necesidad de instalarlos.

La descarga e instalación de los ficheros de voz se inicia mediante la llamada al *intent* TextToSpeech.Engine.ACTION_INSTALL_TTS_DATA ("API", 2014) que redirige al usuario a Google Play, permitiéndole así iniciar la descarga. La instalación ocurre de modo automático una vez la descarga termina.

El constructor de la clase android.speech.tts.TextToSpeech tiene como parámetros de entrada una referencia a la clase Context ("API", 2014) y una referencia a la interfaz TextToSpeech.OnInitListener ("API", 2014), que en el ejemplo de la Figura 2.22, es la propia actividad para ambos casos. Es necesario que la actividad implemente la interfaz TextToSpeech.OnInitListener, de manera que se notifique a la aplicación que el motor TTS ha sido inicializado mediante el método onInit(int status) de dicha interfaz.

2.3.2.3. Configuración del motor de síntesis: selección del idioma

La clase android.speech.tts.TextToSpeech proporciona varios métodos que ayudan a seleccionar el idioma en el que se escuchará la cadena de texto (Trivi, 2009).

- int isLanguageAvailable(Locale locale): permite solicitar la disponibilidad de un idioma para un determinado motor de síntesis. La clase Locale ("API", 2014) permite especificar un país y un idioma a través del constructor Locale(String language, String country)[5]. El código de respuesta puede almacenar el valor TextToSpeech.LANG_COUNTRY_AVAILABLE (disponibilidad del país e idioma especificados), TextToSpeech.LANG_AVAILABLE (disponibilidad únicamente del idioma especificado), TextToSpeech.LANG_NOT_SUPPORTED (ni el idioma ni el país están disponibles), y TextToSpeech.LANG_MISSING_DATA (los recursos del idioma solicitado no están instalados y en este caso la aplicación redirige al usuario a Google Play para encontrar los ficheros correspondientes). Se puede especificar un tercer parámetro a la clase Locale a través del constructor Locale(String language, String country, String variant). Dicho parámetro especifica una variante del idioma[6]. En este caso, la llamada devuelve TextToSpeech.LANG_COUNTRY_VAR_AVAILABLE en el caso de que el idioma, el país y la variante estén disponibles.

- int setLanguaje(Locale locale): permite seleccionar el idioma del motor TTS. El parámetro de entrada es una referencia a la clase Locale y devuelve los mismos códigos de retorno que el método isLanguageAvailable(Locale locale).

Para obtener el valor de la clase Locale que tiene por defecto el dispositivo, se llama al método Locale.getDefault(). Por otro lado, el método getLanguage() de la clase android.speech.tts.TextToSpeech devuelve el valor de la clase Locale que tiene actualmente el motor TTS ("API", 2014).

La Figura 2.23 (Trivi, 2009) muestra un ejemplo en el que se selecciona como idioma del motor de síntesis el español hablado en España. En primer lugar, se comprueba si está disponible para el motor predeterminado, y de estarlo, se establece como idioma del motor de síntesis.

```
if(mTts.isLanguageAvailable(new Locale("spa","ESP")) >= 0){
    mTts.setLanguage(new Locale("spa","ESP"));
}
```

Figura 2.23: Código desarrollado para seleccionar el idioma español hablado en España

Es recomendable configurar el idioma del motor de síntesis dentro del método onInit(int status) de la actividad, que como se ha explicado, se invoca una vez que el motor de síntesis está inicializado.

[5]El idioma se representa mediante un código de dos o tres letras en minúscula definido por la norma ISO 639-1 (Ej. "spa", "en", etc). El país se representa mediante un código de 2 o 3 letras en mayúscula definidos por las normas ISO 3166-1 alpha-2 o ISO 3166-1 alpha-3 respectivamente(Ej. "ESP", "US", etc) (Trivi, 2009).

[6]El código para la variante del idioma no esta especificado en ninguna norma.

2.3.2.4. Reproducción de una cadena de texto con el sintetizador de voz

Una vez que se ha inicializado y configurado el objeto android.speech.tts.TextToSpeech, el sintetizador de voz puede reproducir cualquier cadena de texto a través de la llamada al método speak(String text, int queueMode, HashMap<String, String> params) por parte de dicho objeto.

El motor TTS gestiona una cola que contiene todas las cadenas de texto que esperan a ser sintetizadas, denominadas en términos TTS, elocuciones (*utterances*, en inglés). Toda instancia de la clase android.speech.tts.TextToSpeech puede decidir si la presente elocución interrumpe a la que está siendo sintetizada en el momento de su llegada a la cola, o bien si se debe colocar en la cola y esperar su turno. Dicho comportamiento viene especificado por el segundo parámetro del método speak(String text, int queueMode, HashMap<String, String> params). Si se desea que la solicitud de dicho método interrumpa cualquier síntesis de texto a voz que se estuviese efectuando, se debe pasar como segundo parámetro el modo TextToSpeech.QUEUE_FLUSH. Por el contrario, si se desea que la solicitud de dicho método espere en la cola a que llegue su turno se deberá pasar como segundo parámetro el modo TextToSpeech.QUEUE_ADD (Trivi, 2009).

El tercer parámetro del método speak(String text, int queueMode, HashMap<String, String> params) permite pasar al motor TTS parámetros opcionales, que se especifican a través del par clave/valor de una estructura HashMap ("API", 2014). Si no se quiere especificar ninguna opción, este parámetro se pone a *null*. Para especificar el par clave/valor de la estructura HashMap, la instancia de la clase HashMap invoca al método put(K key, V value), al que se le pasa como primer parámetro la opción que se desea especificar, y como segundo parámetro el valor que toma dicha opción (Trivi, 2009). A continuación se describen algunas de las posibilidades que ofrecen los parámetros opcionales.

Opción TextToSpeech.Engine.KEY_PARAM_STREAM

Se trata de una de las posibilidades para la clave de la estructura HashMap que indica al motor TTS el canal de salida de audio al que dirigir la voz sintetizada. Los posibles valores que puede tomar dicha opción aparecen en la Tabla 2.9 (Hashimi et al., 2010). La Figura 2.24 (Trivi, 2009) muestra como se utiliza la opción KEY_PARAM_STREAM.

Salida de Audio	Descripción
STREAM_ALARM	La salida de audio para alarmas.
STREAM_MUSIC	La salida de audio para la audición de música.
STREAM_NOTIFICATION	La salida de audio para las notificaciones.
STREAM_RING	La salida de audiopara el sonido de llamadas del teléfono.
STREAM_SYSTEM	La salida de audio para sonidos del sistema.
STREAM_VOICE_CALL	La salida de audio para llamadas de teléfono.

Tabla 2.9: Posibilidades de la opción TextToSpeech.Engine.KEY_PARAM_STREAM

```
HashMap<String, String> myHashAlarm = new HashMap<String, String>();
myHashAlarm.put(TextToSpeech.Engine.KEY_PARAM_STREAM,
        String.valueOf(AudioManager.STREAM_ALARM));
mTts.speak(myText1, TextToSpeech.QUEUE_FLUSH, myHashAlarm);
mTts.speak(myText2, TextToSpeech.QUEUE_ADD, myHashAlarm);
```

Figura 2.24: Código utilizado para definir la opción KEY_PARAM_STREAM

Tal y como se observa en en el ejemplo, la instancia de la clase HashMap invoca al método put(K key, V value) especificándose como clave KEY_PARAM_STREAM y como valor STREAM_ALARM con

la intención de respetar los ajustes de alarma que el usuario seleccionó para el dispositivo. Posteriormente, la instancia android.speech.tts.TextToSpeech invoca al método speak(String text, int queueMode, HashMap<String, String> params) al que se le pasa como tercer parámetro la instancia de la clase HashMap. La primera llamada a dicho método interrumpe cualquier otra síntesis de texto a voz que se estuviese efectuando ya que tiene especificado como segundo parámetro el modo TextToSpeech.QUEUE_FLUSH. La segunda solicitud de síntesis de texto a voz se realiza una vez se termine de realizar la síntesis de texto a voz del primer texto ya que tiene especificado en el segundo parámetro el modo TextToSpeech.QUEUE_ADD.

Opción TextToSpeech.Engine.KEY_PARAM_UTTERANCE_ID

En ocasiones, interesa saber el instante en el que ha finalizado la síntesis de una cadena de texto. Esto resulta útil si se quiere realizar una acción inmediatamente después de haberse realizado la síntesis de texto a voz. Para ello, es necesario identificar a la cadena de texto mediante la opción KEY_PARAM_UTTERANCE_ID.

Para alcanzar el propósito que se persigue es necesario que la actividad de la aplicación implemente la interfaz TextToSpeech.OnUtteranceCompletedListener.

De este modo, cuando la instancia de la clase android.speech.tts.TextToSpeech invoca al método setOnUtteranceCompletedListener(TextToSpeech.OnUtteranceCompletedListener listener), el parámetro de entrada hace referencia a la propia actividad. Como consecuencia, cada vez que finaliza una síntesis de texto a voz, se invoca al método onUtteranceCompleted(String uttId) de la clase TextToSpeech.OnUtteranceCompletedListener al que se le pasa como parámetro el identificador de la cadena de texto que acaba de ser sintetizada. La Figura 2.25 (Trivi, 2009) muestra como se utiliza la opción KEY_PARAM_UTTERANCE_ID.

```
mTts.setOnUtteranceCompletedListener(this);
myHashAlarm.put(TextToSpeech.Engine.KEY_PARAM_STREAM,
        String.valueOf(AudioManager.STREAM_ALARM));
mTts.speak(myText1, TextToSpeech.QUEUE_FLUSH, myHashAlarm);
myHashAlarm.put(TextToSpeech.Engine.KEY_PARAM_UTTERANCE_ID,
        "ID del mensaje de alarma");
// myHashAlarm ahora contiene dos parámetros opcionales
mTts.speak(myText2, TextToSpeech.QUEUE_ADD, myHashAlarm);
```

Figura 2.25: Código utilizado para identificar una elocución mediante parámetros opcionales

Tal y como se observa en en el ejemplo, la instancia de la clase HashMap invoca al método put(K key, V value) especificándose como clave KEY_PARAM_STREAM y como valor STREAM_ALARM. A continuación, se pasa dicha instancia como tercer parámetro del método speak(String text, int queueMode, HashMap<String, String> params).

Posteriormente, el objeto HashMap invoca de nuevo al método put(K key, V value), que con el fin de identificar a la cadena de texto, tiene como clave la opción KEY_PARAM_UTTERANCE_ID, y como valor el identificador de la cadena que se envía al motor de síntesis. Dicha instancia de la clase HashMap se pasa como tercer parámetro del método speak(String text, int queueMode, HashMap<String, String> params), de manera que el motor TTS notifica a la aplicación de la finalización de la síntesis de texto a voz de cada cadena.

Utilizando identificadores únicos para cada cadena de texto, se identifica con exactitud la cadena de texto que acaba de ser sintetizada, de manera que siempre que se retorne de cualquier interrupción se pueda partir de la cadena de texto siguiente a la última sintetizada.

La Figura 2.26 (Trivi, 2009) muestra un ejemplo de implementación del método onUtteranceCompleted(String uttId), al que se le envía como parámetro el identificador de la cadena de texto que acaba

de ser sintetizada.

```
public void onUtteranceCompleted(String uttId) {
    if (uttId == "ID del mensaje de alarma") {
        SonidoAlarma();
        }
}
```

Figura 2.26: Código utilizado para implementar el método onUtteranceCompleted

El método setOnUtteranceCompletedListener(TextToSpeech.OnUtteranceCompletedListener listener) no se utiliza en el API de nivel 15 o superior por lo que si se diseña una aplicación de API de nivel 15 o superior se debe utilizar el método setOnUtteranceProgressListener(UtteranceProgressListener) ("API", 2014) en su lugar.

2.3.2.5. Guardar el resultado de la síntesis de texto a voz en un fichero

Ocasionalmente, interesa grabar el resultado de la síntesis de texto a voz en un fichero. Esto es de utilidad cuando el texto va a ser reproducido con frecuencia ya que en lugar de realizar la síntesis de texto a voz cada vez que se quiere escuchar el texto, bastará con reproducir el fichero de voz consiguiendo reducir el consumo de recursos y aumentar la rápidez de la aplicación. Para que la grabación del resultado de la síntesis de texto a voz sea posible, se ha de añadir el permiso android.permission.WRITE_EXTERNAL_STORAGE en el fichero AndroidManifest.xml del proyecto (Hashimi et al., 2010), (Trivi, 2009).

Tal y como muestra el ejemplo de la Figura 2.27 (Trivi, 2009), con el objetivo de que el resultado de la síntesis se guarde en un fichero, la instancia android.speech.tts.TextToSpeech debe invocar al método synthesizeToFile(String text, HashMap<String, String> params, String filename), que al igual que el método speak(String text, int queueMode, HashMap<String, String> params), permite especificar un identificador para informar a la aplicación sobre el instante en el que se guarda el resultado de la síntesis de texto a voz en el fichero. El primer parámetro del método synthesizeToFile(String text, HashMap<String, String> params, String filename) es la cadena de texto a ser sintetizada, el segundo parámetro es la estructura HashMap con el identificador de dicha cadena de texto y el tercer parámetro es el fichero en el que se guarda el resultado de la síntesis.

```
HashMap<String, String> myHashRender = new HashMap<String, String>();
String textoAlarma = "Despiértate";
String destFileName = "/sdcard/myAppCache/wakeUp.wav";
myHashRender.put(TextToSpeech.Engine.KEY_PARAM_UTTERANCE_ID, textoAlarma);
mTts.synthesizeToFile(textoAlarma, myHashRender, destFileName);
```

Figura 2.27: Código utilizado para guardar el resultado de la síntesis de texto a voz en un fichero de audio

Una vez se haya notificado de la finalización de la síntesis, se puede reproducir el fichero de audio del mismo modo que cualquier otro recurso audio, mediante el uso de android.media.MediaPlayer ("API", 2014).

Existen otras dos posibilidades para indicar a la instancia android.speech.tts.TextToSpeech que asocie el contenido de una cadena de texto a un fichero. La primera de las posibilidades consiste en que dicha instancia invoque al método addSpeech(String text, String filename), especificándose en el segundo parámetro la dirección del fichero. La otra opción es identificar el fichero a través de su identificador y del paquete en el que se encuentra mediante la llamada al método addSpeech(String

text, String packagename, int resourceId). Los ficheros de audio se almacenan en el directorio */res/raw* de la aplicación y se referencian mediante su identificador. En ambos casos, es necesaria una base de datos para asociar cada cadena de texto a su fichero correspondiente (Hashimi et al., 2010), (Trivi, 2009).

Al asociar la cadena de texto al fichero, en cada llamada al método speak(String text, int queueMode, HashMap<String, String> params) al que se le pase como primer parámetro una cadena asociada a un recurso de audio, se abre directamente el fichero correspondiente en lugar de hacer la síntesis de texto a voz. Si el fichero no se encontrase, se realizaría la síntesis de texto a voz.

Otra de las utilidades de grabar el resultado de la síntesis de texto a voz en un fichero, es que permite especificar al diseñador la pronunciación de determinadas cadenas de texto. Para ello, se guarda la nueva pronunciación en un fichero que se asocia a la cadena de texto.

Cabe mencionar que el método synthesizeToFile(String text, HashMap<String, String> params, String filename) únicamente guarda el resultado de la síntesis de texto a voz en ficheros con formato WAV, a pesar de la extensión que se dé al nombre del fichero destino. Sin embargo el método addSpeech(String text, String filename) permite asociar una cadena de texto a ficheros de diferentes formatos (Ej. ficheros MP3).

Una limitación que tiene la asociación de una cadena de texto a un recurso audio, es que no sirve si la cadena está contenida dentro la cadena de texto que se pasa al método speak(String text, int queueMode, HashMap<String, String> params). Una solución es dividir el texto en palabras y enviar cada palabra por separado al motor TTS a pesar de que el texto pudiese escucharse entrecortado. Como consecuencia de dicha limitación, la opción de grabar en ficheros audio el resultado de la síntesis de texto a voz únicamente tiene utilidad si se sabe de antemano las cadenas de texto que van a ser reproducidas (Hashimi et al., 2010).

2.3.2.6. Añadir Earcons a la aplicación

Un *earcon*, o icono acústico, es un sonido breve y distintivo usado para representar un determinado evento. Para añadir un earcon, la instancia android.speech.tts.TextToSpeech puede invocar o bien al método addEarcon(String earcon, String filename), que funciona de forma parecida al método addSpeech(String text, String filename), o bien al método addEarcon(String earcon, String packagename, int resourceId), que funciona de forma parecida al método addSpeech(String text, String packagename, int resourceId). El primer argumento en ambos casos es el nombre del earcon, que deberá estar contenido entre corchetes (E.j, "[boing]") .

Para reproducir un earcon se debe invocar al método playEarcon(String earcon, int queueMode, HashMap<String, String> params) cuyos parámetros son los mismos que los del método speak(String text, int queueMode, HashMap<String, String> params) (Hashimi et al., 2010).

2.3.2.7. Incluir una pausa entre dos síntesis de texto a voz

El método playSilence(long durationInMs, int queueMode, HashMap<String, String> params) de la clase android.speech.tts.TextToSpeech se utiliza para forzar una pausa en el motor de síntesis. El primer parámetro representa el número de milisegundos de la pausa, el segundo parámetro representa el modo de la cola que gestiona el motor TTS y podrá contener los mismos valores que el parámetro análogo del método speak(String text, int queueMode, HashMap<String, String> params), y el tercer parámetro especificará los parámetros opcionales a través del par clave/valor de una estructura HashMap.

El método playSilence(long durationInMs, int queueMode, HashMap<String, String> params) se suele emplear con el modo QUEUE_ADD para forzar una pausa entre dos cadenas de texto que se van a escuchar. Para ello, se invoca a dicho método entre dos llamadas al método speak(String text, int queueMode, HashMap<String, String> params) (Hashimi et al., 2010).

2.3.2.8. Finalizar el motor de síntesis

La funcionalidad de síntesis de texto a voz esta compartida entre todas las aplicaciones que la utilizan. Por ello, es conveniente, que una vez que la aplicación haya terminado de utilizar dicha funcionalidad, la instancia de android.speech.tts.TextToSpeech invoque el método stop() dentro del

método onDestroy() de la actividad tal y como se muestra en la Figura 2.28. La llamada a shutdown() libera todos los recursos utilizados por el motor TTS ("API", 2014), (Trivi, 2009).

```
public void onDestroy(){
    if(mTts != null){
        mTts.stop();
        mTts.shutdown();
    }
    super.onDestroy();
}
```

Figura 2.28: Código utilizado para finalizar el motor de síntesis

Si el objeto android.speech.tts.TextToSpeech llama al método stop() antes de la llamada a shutdown() se interrumpe la cadena de texto que esté siendo en ese instante procesada, y se descarta las restantes de la cola ("API", 2014).

2.3.2.9. Otros métodos útiles de la clase TextToSpeech

Otros métodos de la clase android.speech.tts.TextToSpeech que resultan útiles a la hora de diseñar una aplicación que integre TTS son (Hashimi et al., 2010):

- setPitch(float pitch): Especifica el tono de la voz que podrá ser más agudo o más grave según lo especifique el parámetro de entrada. El valor normal es 1.0, el valor más bajo es 0.5 y el valor más alto es 2.0. La velocidad no varía.
- setSpeechRate(float rate): Especifica la velocidad de la voz. El rango de valores oscila entre 0.5 (mínima velocidad) y 2.0 (máxima velocidad), siendo 1.0 el valor normal.
- isSpeaking(): Devuelve *true* o *false* dependiendo de si el motor *TTS* está reproduciendo o no una cadena de texto, incluyéndose la pausa establecida por el método playSilence(long durationInMs, int queueMode, HashMap<String, String> params).

Otra utilidad que ofrece Android para la síntesis de texto a voz, es la de poder notificar a la aplicación una vez que el motor de síntesis ha terminado de procesar todas las cadenas de texto de su cola. Esto se logra mediante la implementación de un BroadcastReceiver que responde a la acción TextToSpeech.ACTION_TTS_QUEUE_PROCESSING_COMPLETED. Para obtener información sobre otros métodos de la clase android.speech.tts.TextToSpeech se puede consultar el API de Android.

2.3.3. Estudio de las alternativas que ofrece Android como motores de síntesis de voz

Todo dispositivo Android incorpora por lo general un motor TTS instalado por defecto, mediante el cual las aplicaciones son capaces de leer textos en voz alta. Sin embargo, Android ofrece la posibilidad de instalar y personalizar varios motores, aunque siempre eligiendo uno como motor TTS principal (Hernández, 2012).

En la Sección 2.3.1 se estudió el motor de síntesis de texto a voz Pico TTS, que viene instalado por defecto en la mayoría de dispositivos Android. En la presente sección se estudiarán los diferentes sintetizadores de texto a voz que ofrece Android, con el fin de seleccionar el que mejores prestaciones ofrezca.

Para seleccionar el motor TTS que se quiere como principal hay que ir al menú de Ajustes->Entrada y salida de voz->Ajustes de síntesis de voz de la Figura 2.20. La Figura 2.29, muestra la parte inferior de dicho menú, en el que aparece una lista con los motores que están instalados en el dispositivo. Para activar un motor de síntesis se ha de seleccionar la casilla correspondiente. Además, se observa que algunos motores constan de un menú de ajustes que permite configurar opciones de personalización

del motor. En la Figura 2.29 aparecen activados todos los motores que se encuentran instalados en el dispositivo. Finalmente, se selecciona el motor de síntesis que se quiere como principal seleccionando la opción Ajustes->Entrada y salida de voz->Ajustes de síntesis de voz->Motor predeterminado. Tal y como se observa en la Figura 2.30, dicha opción muestra una lista con todos los motores de síntesis que se han activado previamente. Aparece el motor *IVONA Text-to-Speech HQ* seleccionado como motor principal.

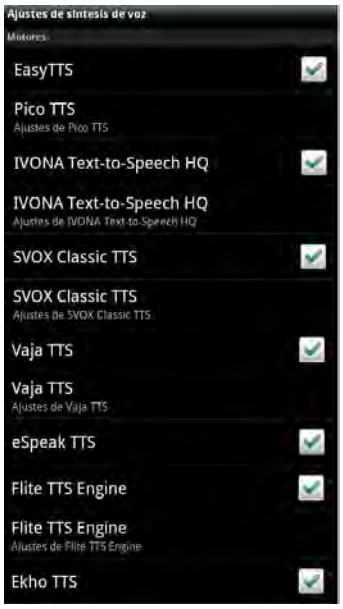


Figura 2.29: Menú de activación del motor de síntesis

Figura 2.30: Pestaña de selección del motor de síntesis predeterminado

Para escuchar un ejemplo del motor de síntesis seleccionado, se selecciona la opción Ajustes->Entrada y salida de voz->Ajustes de síntesis de voz->Escuchar un ejemplo.

La Tabla 2.10 (Hernández, 2012), ("TTS Engines", 2012) resume las características principales de los motores de síntesis que ofrece Android. Estas serán las características que se tengan en cuenta a la hora de seleccionar el motor de síntesis que mejor se integre en la aplicación que se quiera diseñar.

Sintetizador TTS	Idiomas Disponibles para Android	Calidad de las voces	Tamaño descarga	Disponibilidad	Precio
Pico TTS/Google TTS	Inglés EEUU/UK, francés, italiano, alemán y español.	Regular. Monótona y robótica.	Se encuentra instalado por defecto.	Limitada. Únicamente instalado en ciertos dispositivos.	Gratuito.
IVONA TTS HQ	Inglés UK/EEUU/Australiano/Gales, Polaco, Alemán, Francés, Rumano, Italiano, Español Europeo/EEUU, Islandés. En español tiene la voz de Conchita.	Muy buena. Voz clara, natural y con buen acento.	~250MB/voz.	*IVONA TTS HQ* y sus voces pueden descargarse en *Google Play* (Android 1.6 o +).	Gratuito (fase beta).
SVOX Classic TTS	40 voces femeninas y masculinas en más de de 25 idiomas. Supera el número de idiomas de cualquier otro motor *TTS*. En español europeo: voz de Noelia y Pablo.	Muy buena. Voz clara, natural y con buen acento. Herramienta de corrección de la pronunciación.	~10-20MB/voz.	*SVOX Classic TTS* y sus voces pueden descargarse en *Google Play* (version: varía según el dispositivo).	2'99 euros/voz.
Samsung TTS	Coreano, inglés, chino y español. Efectos: profunda, alta, fina, gruesa, robótica, Helio.	Regular. (Similar a la del motor *Pico TTS*)	Instalado por defecto.	Limitada. Únicamente instalado en Samsung.	Gratuito.
CereProc Ltd	Inglés con diferentes acentos: Irlandés (Caitlin), Inglés Midlands(Sue), inglés EEUU(Adam, Isabella y Katherine), escocés(Stuart y Heather), Inglés del sur (Jack). (Español no disponible para Android)	Muy buena. Caitlin comparable a SVOX Victoria UK y a Ivona Amy UK. Adam es la mejor voz para inglés americano.	~140-170 MB/voz	Las voces de *CereProc Ltd* pueden descargarse en *Google Play* (Android 2.3.3 o +).	1.46 euros/voz.
eSpeak TTS	60 idiomas instalados. (Español disponible para Android)	Voz muy robótica y casi incomprensible. Problemas con acentos y reproducción.	1,4 MB (voces inclusive)	*D*isponible en *Google Play* (Android 2.2 o +).	Gratuito.
EasyTTS	27 idiomas instalados. (Español disponible para Android).	Regular. (Similar a la del motor *Pico TTS*)	925KB (voces inclusive).	*D*isponible en *Google Play* (Android 1.6 o +).	Gratuito.
Flite TTS	Está disponible en idioma inglés, con 12 variaciones.	Regular. Entrecortada.	3MB/voz	Disponible en Google *Play* (Android 2.3 o +).	Gratuito.
Loquendo TTS	Susan (Inglés EEUU) y Paola (Italiano).	Muy buena. Incorpora emociones a las voces (voz muy humana).	Desconocido.	No Disponible en *Google Play* actualmente.	Desconocido
Ekho TTS	Cantonés y mandarín.	Desconocido.	490KB. Paquete del datos: 4,65MB.	Disponible en *Google Play*. Paquete de datos se puede descargar. No funciona para Android 4.	Gratuito.
Vaja TTS	Inglés y tailandés.	Desconocido. El paquete de datos parece no estar disponible.	2.6MB.	Disponible en Google Play (Android 2.2 o +). Paquete de datos no disponible.	Gratuito. (versión Beta)

Tabla 2.10: Características principales de los sintetizadores de texto a voz disponibles para dispositivos móviles Android

En los próximos apartados se hace una descripción detallada de las alternativas que ofrece Android al motor Pico TTS, del que ya se habló con detalle en la Sección 2.3.1.

2.3.3.1. IVONA TTS HQ

Sintetizador de voz creado por IVONA (`http://www.ivona.com/en/`) que ofrece multitud de voces femeninas y masculinas cuya calidad es superior a la ofrecida por el sintetizador de voz Pico TTS gracias a la tecnología *BrightVoice TM*.

IVONA permite seleccionar entre un amplio abanico de idiomas[7], que pueden ser testeados en su propia página. En Español, dispone de Conchita y Penélope (esta última, con acento de EEUU). También dispone de Enrique, voz masculina, aunque parece que aún no está disponible para Android. La calidad de la voz es muy buena en todos los idiomas, tratándose de una voz clara, natural y con buen acento, lo que le convierte en uno de los mejores sintetizadores de voz que existe para Android.

El sintetizador IVONA TTS HQ se descarga en Google Play. Tanto IVONA TTS HQ como sus voces se pueden descargar e instalar sin ningún tipo de coste temporalmente, ya que se encuentra en fase beta.

Una vez instalado el sintetizador de voz IVONA TTS HQ, se elige la voz que utilizará el sintetizador de texto a voz y se descarga. En Español se descargará la voz *IVONA Conchita Spanish beta* ("GooglePlay", 2014). Si se han descargado varias voces se puede seleccionar cualquiera de ellas desde el menú de Ajustes->Entrada y salida de voz->Ajustes de síntesis de voz->Ajustes de *IVONA Text-to-Speech HQ*.

En la Figura 2.31 se observa una captura de pantalla de la aplicación IVONA Conchita Spanish beta con la que se podrá probar la voz. Como se observa, la aplicación IVONA Conchita Spanish beta tiene la opción de fijar el tono y la velocidad de la voz.

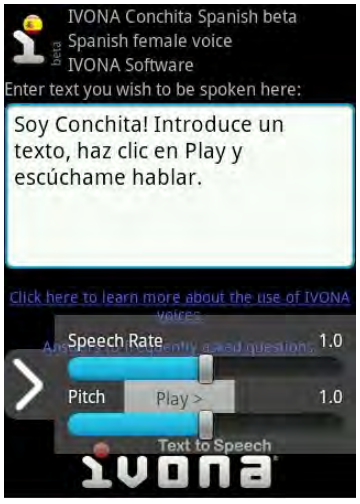


Figura 2.31: Captura de pantalla de la aplicación IVONA Conchita Spanish

2.3.3.2. SVOX Classic TTS

Sintetizador de texto a voz creado por la compañía SVOX (`http://svoxmobilevoices.wordpress.com/`), una de las empresas con más tiempo y experiencia en el sector.

El motor de síntesis SVOX Classic TTS consta de más de 40 voces femeninas y masculinas en más de 25 idiomas[8], superando el número de idiomas de cualquier otro motor de síntesis. Al igual que IVONA, la calidad de las voces es muy buena. En español europeo tiene la voz de *SVOX Spanish Noelia Voice* ("GooglePlay", 2014) y *SVOX Spanish Pablo Voice* ("GooglePlay", 2014), ambas dos de

[7]Idiomas disponibles en: `https://play.google.com/store/apps/developer?id=IVONA+Text-to-Speech`.
[8]Idiomas disponibles en: `https://play.google.com/store/apps/developer?id=SVOX+Mobile+Voices`.

gran calidad. Además incorpora una herramienta que permite corregir la pronunciación de las palabras malsonantes.

La aplicación SVOX Classic TTS se puede descargar de Google Play. Igualmente, se pueden descargar de Google Play las más de 40 voces disponibles para el motor, con un coste de 2'99 euros cada una, lo que supone el principal inconveniente de SVOX Classic TTS. Sin embargo, existe la posibilidad de instalar las versiones de prueba desde Google Play, para probarlas durante dos semanas antes de comprarlas. Además, la propia página de SVOX permite probar algunas demos de las voces.

Una vez instalado el sintetizador de voz SVOX Classic TTS, se debe descargar las voces que utilizará el sintetizador de texto a voz. Para ello, en la aplicación SVOX Classic TTS se selecciona el país y la voz que se desea descargar de Google Play, tal como muestra la Figura 2.32.

Figura 2.32: Menú de selección del idioma en la aplicación SVOX Classic TTS

Al seleccionar la bandera correspondiente al país que se quiere seleccionar, aparecen los paquetes con las voces de dicho país que vienen representados por el nombre de una persona, indicando el sexo de la voz. Al seleccionar una voz, la aplicación permite o bien comprar la voz, o descargar una versión de prueba. Una vez instalada la voz, aparecerá en el menú de aplicaciones del dispositivo como una aplicación diferente.

Dentro del menú Ajustes->Entrada y salida de voz->Ajustes de síntesis de voz->Ajustes de SVOX Classic TTS que se muestra en la Figura 2.33, aparecen las opciones asociadas al motor SVOX Classic TTS que se enumeran a continuación.

Figura 2.33: Menú de ajustes de la aplicación SVOX Classic TTS

- *SVOX TTS Volume*: Aumenta o disminuye el volumen de los mensajes *TTS* entre los valores 0-400 %. El valor por defecto es 100 %. Si se aumenta demasiado el volumen, se introduce distorsión.
- *SVOX TTS Pitch*: Aumenta o disminuye el tono de los mensajes entre los valores 50-200 %. El valor por defecto es 100 %.
- *Advanced settings*: Al seleccionar esta entrada se entra en el menú de opciones avanzadas, cuyas opciones se listan a continuación.

- *Languaje Override*: Al seleccionar esta entrada se usará el idioma y la voz por defecto en todas las solicitudes TTS.
- *Use Optimized Engine*: Esta opción mejora el rendimiento (para móviles de altas prestaciones).
- *Polygot*: Esta opción permite que una voz en un determinado idioma pueda pronunciar determinadas palabras en otro idioma.

- *Select preferred voice per languaje*: Lista de todas las voces instaladas. Las voces del Español europeo aparecen al seleccionar la opción *European Spanish voices* de la Figura 2.33. Al haber instalado únicamente la voz de Pablo, esta es la única que aparece en la Figura 2.34.

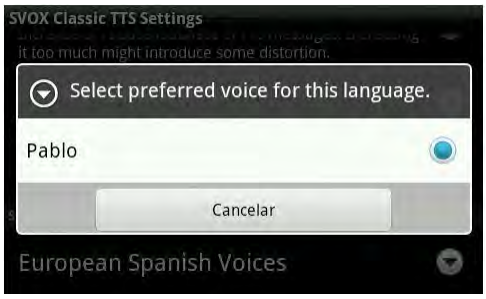


Figura 2.34: Pestaña de selección de la voz en el menú de Ajustes de la aplicación SVOX Classic TTS

En la Figura 2.35 aparece una captura de pantalla de la aplicación SVOX Pablo Spanish. Como se observa, la aplicación permite ajustar el volumen del dispositivo y el volumen, velocidad y tono de la síntesis de texto a voz.

Figura 2.35: Captura de pantalla de la aplicación SVOX Pablo Spanish

La aplicación consta de cinco botones:

- Icono  : Reproduce el texto que se introduce en el campo de texto.
- Icono  : Abre una carpeta con las frases guardadas.
- Icono  : Guarda la frase tecleada en el campo de texto.
- Icono  : Permite compartir la cadena de texto sintetizada con diferentes aplicaciones instaladas en el dispositivo tal como muestra la Figura 2.36.
- Icono  : Abre la herramienta de *Pronunciation correction*. Dicha herramienta se muestra en la Figura 2.37 en la que se a establecido que siempre que aparezca la cadena 'EEUU' se pronuncie como 'Estados Unidos'.

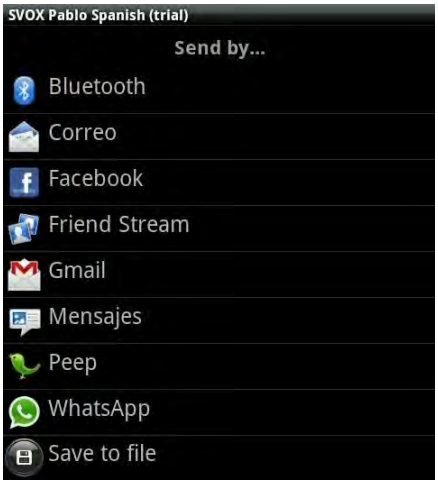


Figura 2.36: Lista de aplicaciones a las que se puede enviar la cadena de texto sintetizada por el sintetizador de voz SVOX Classic TTS

Figura 2.37: Herramienta de corrección de la pronunciación en la aplicación SVOX Classic TTS

2.3.3.3. CereProc Ltd

CereProc (`http://www.cereproc.com/es`) es una empresa escocesa de amplia experiencia en todos los campos relacionados con la tecnología de síntesis de voz. Se dedica a crear soluciones de síntesis de voz para cualquier tipo de aplicación. Su principal producto, *CereVoice*, está disponible para cualquier tipo de plataforma, desde dispositivos incorporados y móviles a ordenadores de sobremesa y servidores.

En Android, CereProc consta de varios sintetizadores de voz: Caitlin (irlandés), Adam, Isabella y Katherine (inglés EEUU), Stuart y Heather (escocés), Jack (inglés del sur), y Sue (inglés *Midlands*)

disponibles en Google Play a 1,46 euros cada uno. Estos sintetizadores son de gran calidad, al estar basados en las soluciones de síntesis de voz que ofrece CereProc para ordenadores de mesa, para las cuales tiene una amplia experiencia. Caitlin es comparable en términos de calidad a SVOX Victoria UK y a Ivona Amy UK. Adam se trata de la mejor voz en cuanto a calidad existente para inglés americano.

También consta de algunas voces gratuitas de poca calidad y utilidad, como *PigLatin voice*, *Dog voice*, *Dodo Glasgow voice*, *Claire Lancashire voice* y *Nicole Sexy French voice* ("GooglePlay", 2014).

Las voces pueden probarse en la página web de CereProc en la parte superior. A pesar de existir una voz en español, ésta no se encuentra disponible para Android por el momento.

2.3.3.4. Samsung TTS

Sintetizador de voz creado por Samsung (`http://www.samsung.com/es/`) basado en los sintetizadores de *HTS Working Group* ("HTS", 2013). Se trata de un sintetizador de voz gratuito que únicamente se encuentra disponible en dispositivos de la marca Samsung. La calidad de la voz es regular, similar a la del sintetizador de voz Pico TTS.

Para seleccionar el motor Samsung TTS como motor predeterminado, se debe entrar desde el dispositivo Samsung en el menú de Ajustes->Entrada y salida de voz->Configuración de síntesis de voz y seleccionar la opción Samsung TTS como motor predeterminado. Se puede seleccionar el idioma del motor de síntesis seleccionando la opción de Ajustes de Samsung TTS en dicho menú.

La Figura 2.38 muestra el menú de Ajustes de Samsung TTS donde se observa que los idiomas instalados son: coreano, inglés, chino y español. Únicamente se encuentran disponibles voces femeninas. La opción Efectos de voz del menú permite activar ciertos efectos para la voz: normal, profunda, alta, fina, gruesa, robótica, o con efecto de Helio.

Figura 2.38: Menú de Ajustes de Samsung TTS

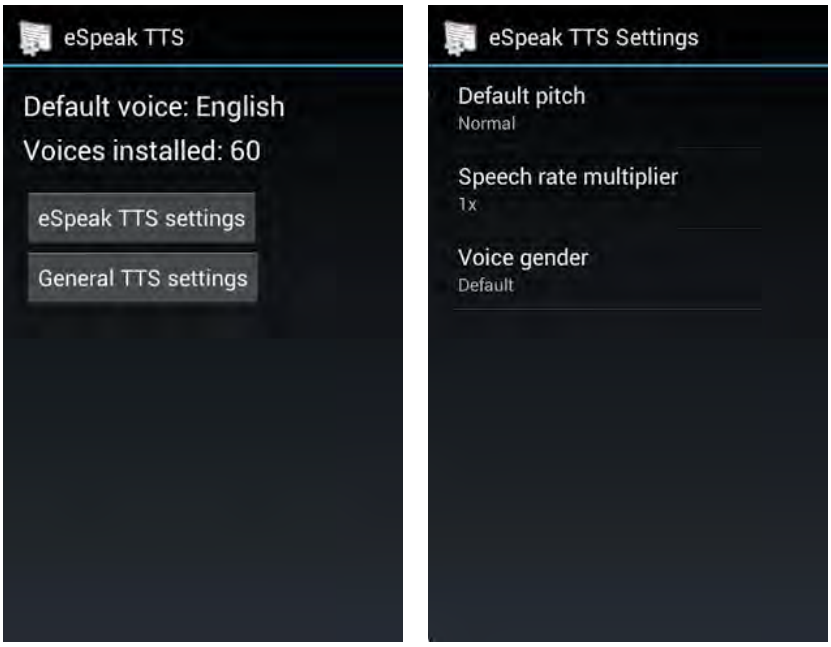
2.3.3.5. eSpeak TTS

Sintetizador de texto a voz basado en el software de código abierto *eSpeak* ("eSpeak", 2014), diseñado por *Eyes-Free Project* para funcionar en plataformas Android. La aplicación eSpeak TTS se puede descargar en Google Play.

El sintetizador eSpeak utiliza como método la síntesis por formantes, por lo que su calidad es inferior a la de otros sistemas de síntesis debido a la reducción de la información, sin embargo, esto permite proporcionar numerosos idiomas ocupando poco espacio.

La Figura 2.39 muestra capturas de pantalla de la aplicación eSpeak TTS. La Figura 2.39.a muestra el menú principal de la aplicación. La opción *General TTS setting* permite seleccionar el idioma del motor de síntesis entre 60 idiomas diferentes. La opción *eSpeak TTS settings* dirige al menú mostrado en la Figura 2.39.b en el que aparecen tres opciones:

- *Default pitch*: Permite seleccionar el tono de la voz.
- *Speech rate multiplier*: Permite seleccionar la velocidad de la voz.
- *Voice gender*: Permite seleccionar el género de la voz(voz masculina o voz femenina).



(a) Menú principal de eSpeak TTS (b) Ajustes del motor eSpeak TTS

Figura 2.39: Capturas de pantalla de a aplicación eSpeak TTS

La voz es muy robótica y casi incomprensible, aunque puede servir para emular voces robotizadas. Aún tiene algunos problemas de reproducción y de pronunciación de palabras acentuadas.

2.3.3.6. EasyTTS

Sintetizador de texto a voz, basado en eSpeak. Se trata de una versión mejorada de eSpeak TTS ya que la calidad de sus voces es mayor (similar a la del sintetizador Pico TTS). Sin embargo, está disponible en un menor número de idiomas[9].

La aplicación EasyTTS se descarga en Google Play.

La Figura 2.40 muestra una captura de pantalla de la aplicación EasyTTS. Tal y como se observa consta de un campo de texto en el que se introduce el texto que va a ser sintetizado y tres opciones de selección:

- *Settings*: Permite seleccionar entre 27 idiomas diferentes.

[9]Idiomas disponibles en EasyTTS: Alemán, chino, afrikaans, bosnio,checo,galés, griego, inglés, esperanto, español, finés, francés, griego antiguo, hindi, húngaro, indonesio, islandés, italiano, lojban, kurdo, latín, macedonio, neerlandés, noruego, polaco, portugués, rumano, eslovaco, sueco, suahili, tamil, turco, y vietnamita.

- *Help*: Se abre una ventana de ayuda con las instrucciones sobre el modo de empleo de EasyTTS.
- *About*: Se abre una ventana que muestra la dirección de contacto del fabricante y otros datos de la aplicación.

Figura 2.40: Captura de pantalla de a aplicación EasyTTS

2.3.3.7. FLite TTS

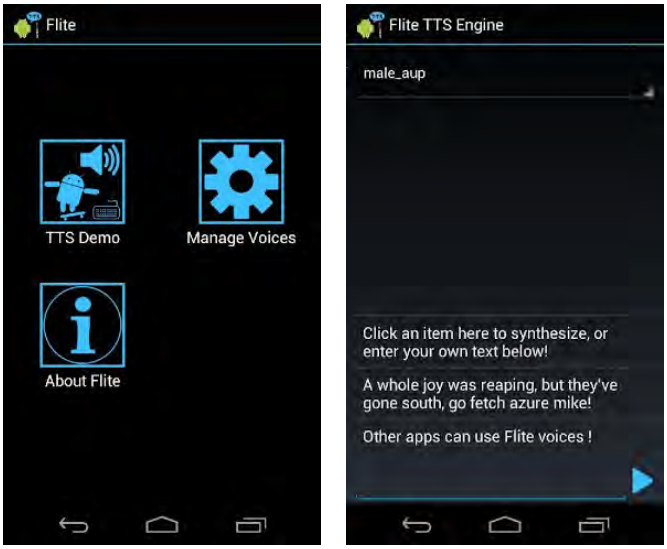
FLite (*Festival Lite*) ("Flite", 2014) es un sintetizador de voz rápido y de poco tamaño desarrollado en *Carnegie Mellon University* diseñado fundamentalmente para sistemas empotrados. Fue originalmente desarrollado en *Festvox* (`http://festvox.org/`) y posteriormente, portado a Android. El código fuente está disponible en *github* ("Flite TTS Engine", 2014). La calidad de la voz es relativamente buena pero no hay versión en español. FLite TTS se encuentra disponible en Google Play de forma gratuita, y dispone de voces en inglés con diferentes acentos[10].

Una vez que se ha seleccionado el motor de síntesis FLite TTS Engine como motor predefinido, han de descargarse las voces seleccionando la opción "Instalar archivos de datos de voz" del menú menú de Ajustes->Entrada y salida de voz->Ajustes de síntesis de voz para que pueda funcionar la aplicación.

La Figura 2.41 muestra capturas de la aplicación FLite TTS. La Figura 2.41.a muestra la venta principal de la aplicación que contiene tres opciones:

- *TTS Demo*: Permite escuchar un ejemplo en cualquiera de las voces descargadas. En el ejemplo de la Figura 2.41.b está seleccionada la opción male_aup (Voz masculina, Inglés India). Escribiendo el texto que se quiere escuchar en el campo de texto inferior y presionando el botón correspondiente, se escucha un ejemplo con el acento correspondiente.
- *Manage Voices*: Abre la ventana que permite la descarga de cualquiera de las 12 voces disponibles para Android.
- *About FLite*: Abre una ventana con información de la aplicación (página web, versión, fabricante, etc).

[10]RMS(Voz masculina, EEUU), BDL(Voz masculina, EEUU), SLT(Voz femenina, EEUU), CLB(Voz femenina, EEUU), JMK(Voz masculina, Inglés canadiense), AWB(Voz masculina, Inglés escocés), AWB(Voz masculina, Inglés Alemania), FEM(Voz masculina, Inglés Alemania), RXR(Voz masculina, Inglés Israel), AUP(Voz masculina, Inglés India), KSP(Voz masculina, Inglés India), y GKA(Voz masculina, Inglés India).



(a) Menú principal de Flite TTS (b) Opción TTS Demo de la aplicación Flite TTS

Figura 2.41: Capturas de pantalla de la aplicación Flite TTS

2.3.3.8. Loquendo TTS

El conocido sintetizador *Loquendo TTS* ("Loquendo TTS", 2014), tiene también su versión para Android. Sin embargo, lleva un tiempo sin estar disponible en Google Play por razones desconocidas, ya que la propia página de Loquendo sigue haciendo referencia a las versiones para Android.

Loquendo dispone de más de 70 voces en más de 30 idiomas, incluyendo también voces en gallego, catalán, valenciano, etc. En español, las voces de Loquendo son tres: Jorge , Carmen y Leonor. Sin embargo para Android solo están disponibles las voces de Susan (Inglés EEUU) y Paola (Italiano). Además, Loquendo utiliza un sistema para reproducir ciertas emociones, como risas, llantos, besos u otros. Se puede escuchar ejemplos en su propia página.

La Figura 2.42 ("Loquendo TTS", 2014) muestra una captura de pantalla de la aplicación Loquendo TTS para Android sacada de su página web. Como se observa, la página web de Loquendo TTS, permite además probar las emociones existentes.

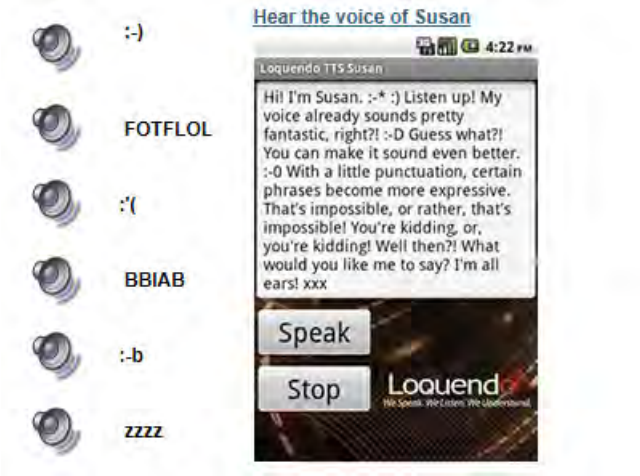


Figura 2.42: Captura de pantalla de la aplicación Loquendo TTS

Loquendo TTS te permite escoger entre dos estilos de lectura:

- *Navigation style*: Permite a Loquendo TTS expandir de forma correcta abreviaciones de direcciones que aparecen comúnmente en ciertas aplicaciones como Google Maps y Google Navigation.

- *Messaging style*: Incluye un decodificador de abreviaciones utilizadas en mensajes de texto, que lee de forma correcta acrónimos y emoticonos comúnmente utilizados en mensajería de texto y emails.

2.3.3.9. Ekho TTS

Sintetizador de texto a voz en chino que funciona tras el proyecto *eGuideDog* ("eGuideDog", 2014), dedicado a desarrollar software gratuito para invidentes.

Ekho TTS puede descargarse de Google Play y permite establecer como idioma del motor de síntesis el cantonés o el mandarín. En el caso de tener problemas con el paquete de idioma de Ekho, puede descargarse directamente[11] y extraerse en la tarjeta SD, en la carpeta /sdcard/ekho/ekho-data/. Actualmente, no funciona en Android 4 (*Ice Cream*) o superiores, ya que se está trabajando en la compatibilidad para estas versiones.

2.3.3.10. Vaja TTS

Vaja TTS (`http://www.vajatts.com/`)es un sintetizador de texto a voz que incluye idiomas inglés y tailandés. Tiene una única voz cuyo nombre es Nok y que habla ambos idiomas.

Una vez que se ha seleccionado el motor de síntesis Vaja TTS como motor predefinido, ha de descargarse el paquete de datos, sin embargo en el momento de escribir el presente estudio parece no estar disponible, por lo que no se pueden probar la voz de Vaja TTS.

La Figura 2.43 muestra una captura de pantalla de la aplicación *Vaja TTS*. Tal y como se observa, la aplicación permite ajustar el volumen, velocidad y tono de la voz.

Figura 2.43: Captura de pantalla de la aplicación Vaja TTS

[11] ekho-data-android-1.0.tar.bz2: Disponible en: `http://dir.eguidedog.net/ekho/ekho-data-android-1.0.tar.bz2`.

2.4. Aplicaciones para dispositivos móviles Android con ASR y TTS

2.4.1. Asistentes de voz para dispositivos móviles Android que incorporan reconocimiento de voz y síntesis de texto a voz

Además de la aplicación de búsqueda por voz de Google, que con la nueva actualización para Android 4.1 Jelly Bean ha pasado a ser un asistente personal parecido al de Siri de Apple, existen otros asistentes de voz para dispositivos móviles Android. La Tabla 2.11 (Rodrigo, 2012), (Pavan, 2013), ("GooglePlay", 2014) resume los principales asistentes de voz existentes en Android que integran reconocimiento de voz. La mayor parte de dichos asistentes de voz integran el reconocimiento de voz de Google para conectarse a los servicios de voz de Google aunque algunos de ellos utilizan servicios de voz propios.

La mayor parte de los asistentes de voz descritos en la Tabla 2.11 integran además síntesis de texto a voz. Gran parte de dichos asistentes, lo único que hacen es utilizar el motor TTS principal (especificado por el usuario) para realizar las lecturas de texto.

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
Assistant (Speaktoit)	Con más de 5 millones de descargas, una puntuación promedio de 4.7 en base a más de 162.000 calificaciones, y elegida como una de las 10 mejores aplicaciones del 2011 según *The New York Times, Speaktoit Assistant* es uno de los mejores asistentes de voz para dispositivos móviles Android. Es un asistente virtual que utiliza lenguaje natural para contestar preguntas, buscar información, iniciar aplicaciones y conectarte con servicios web. Actualmente, esta integrado con los servicios web: Google, Twitter, Facebook, Foursquare, Evernote, Yelp, TripAdvisor, Wikipedia, Chacha, IMDB, Eventful, News360, Amazon, Gmail, Google Images, Google Calendar, Google Maps, Google Navigation, y Waze. La propuesta de Assistant es brindar un asistente inteligente que se encargue de cubrir las necesidades del usuario. ▪ Assistant tiene memoria de las acciones y "recuerda" cuáles son los lugares y servicios predilectos del usuario. Como consecuencia, permite tomar decisiones inteligentes en base al entorno y el comportamiento previo. Así, logra funcionar de manera personalizada. ▪ Gracias a la implementación del lenguaje natural, no es necesario aprender comandos específicos para interactuar con la aplicación, sino que se puede hacer directamente sin riesgo de una incomprensión.	Disponible en Inglés. Español, Ruso, Alemán y Portugués.	Disponible en Google Play para Android 2.1 y superior. Gratuito.

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
	▪ Ofrece asistencia al usuario cuando cree que este lo necesita. ▪ Se trata de un asistente multi-plataforma ya que opera en smartphones, tablets, y portatiles. ▪ Aprende del usuario por lo que el usuario puede mejorar la eficiencia del asistente. Assistant permite realizar un amplio número de tareas. Por ejemplo, mantiene una conversación con el usuario respondiendo a sus preguntas, realiza llamadas telefónicas, envía y recibe notificaciones de mensajes de texto y de correo, configura alarmas, ayuda a encontrar lugares, solicita pronósticos del tiempo, lee noticias, realiza cálculos matemáticos, actualiza estados del Facebook, organiza el calendario, realiza conversiones de monedas, reproduce música, realiza traducciones, etc.		
Sherpa Next	Sherpa es un asistente de voz para dispositivos móviles Android que todavía está en fase beta por lo que se necesita abrir una cuenta la primera vez que se arranca dicho sistema de control por voz, estando limitado su ingreso hasta que los desarrolladores lancen una versión final. Sherpa es uno de los mejores asistentes de voz en español para dispositivos móviles Android. El usuario puede preguntar y obtener respuestas de una gran diversidad de temas, realizar llamadas, enviar SMS y correos, abrir aplicaciones, reproducir música, solicitar información de vuelos, ver resultados deportivos, realizar operaciones matemáticas, actualizar el estado del perfil de redes sociales como Facebook o Twitter, solicitar información sobre la programación, enviar mensajes de WhatsApp, encontrar diferentes tipos de locales, leer noticias, solicitar definiciones o traducciones de palabras, consultar el clima, enviar dinero por PayPal a un contacto, solicitar información bursatil, etc. Para recuperar menciones de Twitter o publicaciones de Facebook se debe incluir los datos de estas redes sociales en Sherpa, pudiendo después publicar directamente con la voz. Una de las características más importantes de Sherpa, es que con la tecnología *Intelligent Proactive System Shertab*, es capaz de aprender los hábitos del usuario por lo que ofrece la información que más le pueda interesar al usuario.	Disponible en español e inglés.	Disponible en Google Play para Android 2.2 y superior. Todavía se encuentra en fase beta. Gratuita.

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
	Además, el módulo informacional *Multiknowledge System* exclusivo de Sherpa incorpora un algoritmo que, a través de una gran variedad de fuentes de información, es capaz de ofrecer resultados muy exactos.		
Vita	Vita es uno de los mejores asistente de voz en español para dispositivos móviles Android. Puede emplearse como herramienta de consulta, entretenimiento o para facilitar el uso del teléfono a diversos colectivos con algún tipo de discapacidad. Incluye funciones como llamar por teléfono, enviar SMS o correos electrónicos, mostrar información completa de contactos, contar chistes, decir horóscopos, consultar operaciones matemáticas, actualizar el estado del perfil de redes sociales como Facebook o Twitter, consultar un cambio de divisas, mostrar información de la hora y la fecha, consultar la predicción del tiempo, lanzar aplicaciones, Geolocalización, búsqueda de lugares/negocios, traductor de idiomas, preguntas de índole general, búsquedas en google y youtube, búsquedas de noticias e imágenes, recetas de cocina, definiciones, fijar alarmas/recordatorios, letras de canciones, adivinanzas, proverbios, cine (cartelera, estrenos, argumentos), activación/desactivación de WIFI/Bluetooth, consulta del nivel de batería, trabalenguas, hora mundial , Aviso de SMS entrantes, Rutas, reproductor de música, búsqueda bíblica, resultados loterías.	Disponible en español.	Disponible en Google Play para Android 2.2 y superior. Todavía se encuentra en fase beta. Gratuita.
Asistente personal por voz	Asistente virtual por voz que pueden llevar a cabo varias tareas, interactuar por voz y responder a los gestos del usuario. Los gráficos realistas en 3D y la cantidad de opciones disponibles hacen que esta aplicación sea única en su género. Dicho asistente es capaz de Despertar el usuario y mostrarle su horóscopo, recordarle eventos importantes, buscar las últimas noticias en las páginas webs más populares (mediante RSS), enviar SMS o realizar llamadas, Mostrar el pronóstico del tiempo (Foreca Weather integrado), buscar casi cualquier tipo de información (Wikipedia integrada), tomar notas o hacer cálculos, ubicar lugares en el mapa y trazar una ruta (Google Maps integrado), ir de compras en línea con Shoptimus o reservar entradas y hoteles con Smartive Hotels, encontrar la información necesaria en Internet a	Disponible en inglés, ruso y español. Las versiones en francés, alemán, japonés y coreano estarán pronto disponibles.	Disponible en Google Play para Android 2.2 y superior. Gratuita.

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
	través de Google, leer las últimas actualizaciones de tus contactos en Facebook o Twitter, sintonizar una emisora de radioo programa de televisión, contar chistes y, hasta bailar. El asistente está disponible en cualquier momento del día o de la noche si quieres charlar o jugar a algo.		
Jeannie	Asistente virtual que permite: Responder preguntas, realizar llamadas, enviar correos, configurar alarmas y recordatorios, escuchar música, escuchar noticias, etc.	Disponible en varios idiomas, incluidos español e Inglés en varios acentos.	Disponible en Google Play para Android 2.2 y superior. Gratuita.
Alicoid	Alicoid es un asistente virtual alemán que también está disponible en español. Al igual que los seres humanos, a Alicoid le gusta que el usuario utilice frases completas en lugar de comandos. Dicho asistente permite investigar hechos así como mandar mensajes cortos, llamar a tus amigos, etc. (Ej. ¿Cómo llego a Sevilla?, ¿Qué tal el tiempo mañana en Madrid?, ¿Quién ha dirigido Amores Perros?, ¿De qué trata Todo Sobre Mi Madre?, ¿Cómo jugó el Real Madrid?, etc.).	Disponible en español y otros idiomas[12], aunque la versión en español aun tiene que mejorar.	Disponible en Google Play para Android 2.2 y superior. Precio: 2.99 euros
Nina (Nuance Interactive Natural Assistant)	Nina es un sistema de reconocimiento de voz que la empresa Nuance, responsable de algunas de las capacidades de Siri (la aplicación para este efecto usada con iOS), ha desarrollado para teléfonos basados en Android y también para iOS. Las principales ventajas de Nina se describen a continuación. ▪ El sistema de reconocimiento de voz que integra Nina permite reconocer la voz del propietario del teléfono gracias a una tecnología que identifica la huella de voz biométrica de la persona. De esta forma, se pueden incorporar diferentes opciones de seguridad. ▪ Proporciona un SDK abierto para ampliar con rapidez las aplicaciones de Android e iOS existentes. ▪ Nina esta disponible en 38 idiomas y permite la personalización del personaje: aspecto, voz, etc.	Disponible en 38 idiomas, entre ellos el español.	Proporciona un kit de desarrollo de software (SDK) abierto diseñado especialmente para las grandes organizaciones empresariales que implementan aplicaciones móviles de atención al cliente.

[12]Idiomas disponibles: alemán, árabe, búlgaro, catalán, chino, checo, danés, holandés, inglés, estonio, finlandés, francés, griego, haitiano, hebreo, hindi, húngaro, indonesio, italiano, japonés, coreano, letón, lituano, noruego, polaco , portugués, rumano, ruso, eslovaco, esloveno, español, sueco, tailandés, turco, ucraniano y vietnamita.

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
	▪ Nina ofrece interacciones de autoservicio naturales e intuitivas: Nina reconoce la voz del usuario y entiende lo que este desea a partir de su forma natural de formular su petición. Nina simplifica el inicio de sesión, la búsqueda de funciones en la aplicación, la formulación de preguntas y la realización de transacciones con ayuda de la voz, la escritura de texto o el punteo en la pantalla. ▪ Entrada/salida multimodal para disfrutar de la máxima flexibilidad mediante la voz, la escritura y el punteo. ▪ Más allá de los simples comandos de voz, Nina entabla una conversación con los clientes: Nina puede ser tan interactivo como el cliente desee ya que permite tanto peticiones sencillas en lenguaje cotidiano como una conversación prolongada con diálogos contextuales y de giros múltiples. El diálogo conversacional requiere un complejo conjunto de tecnologías que se combinan para ofrecer una experiencia de usuario eficaz y atractiva. Nina elimina la complejidad y proporciona interacciones de servicio naturales e intuitivas. Es necesario contar con conexión a Internet para poder hacer uso del sistema, ya que parte del proceso de reconocimiento se realiza en los servidores de la compañía.		
Skyvi	Skyvi es un asistente de voz para dispositivos móviles Android que permite llamar o enviar mensajes a amigos, encontrar lugares, consultar información sobre lugares, actores, deportistas y otras celebridades y actualizar el estado en Facebook y Twitter. Además, también sirve para reproducir música, contar chistes, consultar acerca de la hora y el estado del tiempo y muchas cosas más. Su principal ventaja es su gran precisión para entender lo que dice el usuario, la cual es superior al resto de las aplicaciones.	Unicamente disponible en Inglés.	Disponible en Google Play para Android 2.2 y superior. Gratuita.
Dragon Mobile Assistant	Asistente personal desarrollado por la empresa Nuance que integra reconocimiento de voz y síntesis de texto a voz. Una de las principales carácterísticas de dicho asistente es que se activa mediante el comando de voz "*Hello Dragon*" y a partir de entonces permite ejecutar una amplia variedad de	Unicamente disponible en Inglés.	Disponible en Google Play para Android 2.3.3 y superior. Unicamente disponible en EEUU. Gratuito.

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
	comandos de voz a través de su sistema de reconocimiento de voz. Integra un modo que detecta cuándo el usuario se encuentra en un vehículo en movimiento para hacer saltar automáticamente el Modo Conductor y gestionar el terminal con la voz, pudiendo así centrarse en la carretera. Son muchas las acciones de voz que se pueden realizar mediante Dragon, y que no se distancian demasiado de las acciones que se llevan a cabo gracias a Siri. Por ejemplo, enviar mensajes de texto, escribir un correo electrónico, crear un evento de calendario, hacer actualizaciones a Facebook y Twitter, utilizar los mapas para obtener direcciones, compartir la ubicación, encontrar a amigos en base a su ubicación, discado automático, lectura de mensajes de texto, abrir aplicaciones, realizar llamadas, buscar restaurantes y solicitar reservas, solicitar la previsión del tiempo, escuchar música en el dispositivo, establecer alarmas, etc.		
Sonalight Text by Voice	Asistente virtual que permite realizar llamadas en manos libres y escribir y leer SMS durante la conducción. Ofrece la opción al usuario de responder a los SMS mediante la voz durante la conducción estableciendo la opción de respuesta automática de SMS. Además permite activar el reconocimiento de voz de manera automática durante la conducción. Se ejecuta en segundo plano por lo que se pueden utilizar otras aplicaciones al mismo tiempo.	Unicamente disponible en Inglés	Disponible en Google Play para Android 2.2 y superior.
AIVC(Alice)	Asistente virtual que permite: Realizar llamadas, enviar SMS y correos, Traducir palabras, informar de como ir a determinador lugares, configurar alamas, lanzar aplicaciones, establecer un temporizador, consultar operaciones matemáticas, buscar en la web, actualizar estados en Facebook, consultar la información meterológica, buscar imagenes, etc. Además, permite al usuario definir sus propios comandos y ofrece la oportunidad de controlar dispositivos accesibles desde cualquier interfaz de red. Además, la versión completa de Alice permite configurar eventos en el calendario, reproducir música y videos, controlar la recepcion para Enigma2, establecer el modo dialogo y establecer la opción de que el reconocedor se mantenga a la escucha en segundo plano. Para poder utilizar esta aplicación es necesario tener instalada la aplicación de Busqueda por voz de Google. La síntesis de voz funciona a través de iSpeech.	Inglés y Alemán.	Disponible en Google Play para Android 2.2 y superior. Dispone de varias versiones: La versión completa, AIVC (Alice) - Pro Version, a un precio de 2.99 euros. La version gratuita, AIVC (Alice).

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
EVA - Assistant	EVA es una aplicación que oficia de asistente personal y se integra con las funcionalidades del dispositivo móvil para ofrecer una experiencia completa sin que el usuario tenga que tocar la pantalla. EVA permite al usuario leer y responder mensajes de texto, realizar acciones a través de *Google Voice*, responder correos electrónicos (*Microsoft Exchange* inclusive), responder los mensajes de *Google Talk*, establecer recordatorios según la hora o la ubicación del usuario, realizar acciones según la hora o la ubicación del usuario, establecer el modo de conducción que incorpora una frase de activación, etc. Además dispone de integración con *Google+*, *Evernote*, y *Tasker.*	Unicamente disponible en Inglés.	Disponible en Google Play para Android 2.2 y superior. Dispone de varias versiones: Por un lado, la versión completa, EVA Assistant, y por el otro, su contraparte masculina EVAN. Ambas a un precio de 15.99 euros. Además, dispone de la versión *EVA Intern* que es gratuita.
S Voice (Criado, 2012) , (Jiménez, 2012)	Asistente de voz desarrollado por Samsung que se incorporó por primera vez a los dispositivos Samsung Galaxy S3. Es capaz de controlar la cámara, abrir aplicaciones, realizar llamadas, configurar tareas, configurar alarmas, enviar SMS y correos electrónicos, mostrar la predicción del tiempo, controlar el navegador GPS por medio de la voz, etc. Destaca su capacidad para reconocer la voz natural del usuario. Sin embargo, sigue siendo necesario el estar permanentemente conectados a una conexión de datos para que su funcionamiento sea óptimo. La versión en español se llama Voz S. El comando que ha elegido Samsung para abrir Voz S es “Hola Galaxy” aunque se puede configurar dicha opción. Otra de las características de este asistente de voz es que se puede activar el Voz S incluso con el móvil en reposo. Para eso, no se debe tener activado ningún patrón de bloqueo.	Disponible en inglés UK / EEUU, italiano, alemán, frances, español continental, español latinoamericano, y coreano.	Viene instalada por defecto en los dispositivos Samsung Galaxy S3 y S4. Sin embargo, puede funcionar en cualquier dispositivo con Android 4.0 Ice Cream Sandwich. Para ello, hay que instalar el apk de S voice mediante un explorador de archivos cualquiera. Para instalarla en un terminal Samsung, hace falta ser root del móvil.
Evi	Evi es un asistente de voz para dispositivos móviles Android que trabaja con inteligencia artificial. En lugar de trabajar como un motor de búsqueda, Evi comprende lo que solicita el usuario y brinda información detallada, en lugar de una lista con resultados potenciales. De la misma manera que Siri, Evi permite ejecutar numerosas acciones mediante comandos	Unicamente disponible en Inglés. En lo que se refiere a la información localizada, solamente puede	Disponible en Google Play para Android 2.2 y superior. Gratuita.

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
	de voz, sin necesidad de tocar la pantalla. En primer lugar, se puede hablar con Evi, pedirle algo de información basada en la ubicación, navegar por Internet desde un navegador integrado, etc.	brindarla para Gran Bretaña y Estados Unidos.	
Indigo	Indigo es un asistente virtual multi-plataforma que permite al usuario comenzar una conversación con su smartphone, continuarla en su tablet u ordenador de mesa y finalizarla de vuelta en su smartphone. Esta característica lo convierte en un asistente ideal para usuarios que se estan moviendo. El asistente virtual Indigo permite: Continuar la misma conversación en diferentes dispositivos, buscar en Internet la respuesta a cualquier pregunta, actualizar estados en redes sociales como facebook y Tweeter, escuchar recomendaciones de restaurantes, dirigir al usuario a lugares cercanos, guardar notas y recordatorios, enviar y leer SMS y correos, realizar llamadas, reproducir videos de Youtube, etc.	Únicamente disponible en inglés.	Disponible en Google Play para Android 4.0 y superior. Todavía se encuentra en fase beta. Gratuita.
Cloe	Cloe es el primer asistente por voz en Español para dispositivos móviles. Esta orientado a usarse de forma exclusiva en España por lo que cuenta con un sistema de “Modulos de aprendizaje” con información preparada de antemano. Algunos módulos son: Cambio de divisas, traduccion de expresiones, hora mundial, previsión meteorológica, horarios y tarifas de Renfe, información sobre la liga de fútbol BBVA, programación de TV nacional, letras de canciones, sinopsis de películas, recetas de cocina., etc. Para poder utilizar esta aplicación es necesario tener activada la opción de entrada y salida de voz en Android, disponible en "ajustes del dispositivo".	Unicamente disponible en español.	Ya no se encuentra disponible.
Maluuba International Beta	Asistente virtual que permite: ▪ Configurar recordatorios, alarmas y eventos del calendario. ▪ Realizar preguntas con *Wolfram Alpha* (Buscador de respuestas). ▪ Conectarse a Facebook, Twitter, Foursquare, etc. ▪ Búsqueda de restaurantes y negocios (dependiendo de la ubiación, funciona mediante Yelp).	Únicamente disponible en inglés.	Disponible en Google Play para 2.3.3 o superior. Todavía se encuentra en fase beta. Gratuito.

Aplicación	Descripción y funcionalidades	Idiomas soportados	Disponibilidad y Precio
	▪ Realizar llamadas, enviar SMS y correos a contactos. Dicha lista de características que ofrece Maluuba International varía según el país en el que se ecnuentre el usuario.		
Andy	Andy es un asistente personal para dispositivos Android que reconoce la voz del usuario, busca en Internet la mejor respuesta y la devuelve al usuario mediante una voz clara, además de mostrarla por pantalla. Para realizar la consulta o bien se pulsa el microfono que se muestra por pantalla para activar el reconocedor de voz, o se escribe por teclado la consulta, o se agita el teléfono (para ello es necesario tener habilitada dicha opción). Una de las principales características de Andy es que funciona con auriculares Bluetooth. Andy permite: Realizar preguntas generales, consultar información acerca de celebridades, consultar información de películas, consultar música, abrir aplicaciones Android, buscar en Internet, configurar el calendario, solicitar la ubicación, cambio de divisas, solicitar información del dispositivo, solicitar información de distancias y direcciones a lugares, establecer llamadas, enviar correos y SMS, solicitar información de vuelos, solicitar información de la bolsa, realizar cálculos matemáticos, reproducir música, personalizar el asistente para que se diriga al usuario mediante su nombre, establecer recordatorios y alarmas, solicitar resultados deportivos, buscar imagenes, solicitar información metereólogica, solicitar como se escribe una determinada palabra, etc.	Disponible unicamente en inglés.	Disponible en Google Play para Android 2.2 y superior. Dispone de varias versiones: Por un lado, la versión completa, con un coste de 2.39 euros, y por el otro, la versión limitada siendo esta gratuita.

Tabla 2.11: Asistentes de voz para dispositivos móviles Android

2.4.2. Aplicaciones que integran motores de síntesis de texto a voz

La ventaja de los motores de síntesis reside en su fácil integración en multitud de aplicaciones TTS existentes en Android dotándolas de una mayor utilidad. Es importante saber que la mayor parte de aplicaciones que integran síntesis de voz existentes en Android, lo único que hacen es utilizar el motor TTS principal (especificado por el usuario) para realizar las lecturas de texto.

La integración de motores de síntesis en aplicaciones TTS en Android permite entre otras muchas posibilidades ("GooglePlay", 2014):

- Escuchar en voz alta durante la conducción las direcciones de lugares mediante Google Navigation, sin necesidad de que el conductor lea la pantalla del dispositivo.
- Leer en voz alta eBooks con las aplicaciones *Book Speech* y *ReadBoox*.
- Escuchar mensajes de texto para potenciar la seguridad en la conducción mediante las aplicaciones *Handcent SMS*, *Drive Carefully*, etc.

- Escuchar notificaciones del Facebook y Twitter con la aplicación *iHear Network*.

La Tabla 2.12 ("App Gallery", 2014) resume alguna de las muchas aplicaciones Android que utilizan la síntesis de texto a voz.

Aplicación	Descripción y funcionalidades	Disponibilidad y Precio
Speak Text Easy	Permite escuchar un texto, un libro, en formato epub o txt. Utiliza el motor de síntesis principal especificado por el usuario. La aplicación es capaz de hablar varias lenguas en el mismo texto a condición de que varios lenguajes estén instalados sobre el dispositivo.	Disponible en Google Play para Android 2.2 y superior. Precio: 1,50 euros.
Announcify	Lee la identidad del llamante, los mensajes de texto, email, etc.	Disponible en Google Play. Version requerida: depende del dispositivo. Gratuita.
Talking Caller ID	Reproduce en voz alta el nombre o el número de llamadas entrantes.	Disponible en Google Play para Android 1.6 y superior. Gratuita.
Enhanced SMS & Caller ID	Proporciona notificaciones de voz para las llamadas entrantes, mensajes SMS/MMS, Google Talk, Gmail, los nuevos mensajes K9 / Kaiten / AquaMail y recordatorios de eventos de *Google Calendar*.	Disponible en Google Play. Version requerida: depende del dispositivo. 2,29 euros.
Handcent SMS	Incorpora TTS a la mensajería del dispositivo. Tiene más funcionalidades que la aplicación de mensajes por defecto y la posibilidad de personalización completa.	Disponible en Google Play. Android 1.5 y superior. Gratuita.
ezPDF Reader	Lector de ficheros pdf.	Disponible en Google Play. Android 2.1 y superior. 3,20 euros.
PDF a voz	Reproduce archivos PDF y archivos de texto (txt).	Disponible en Google Play. Version requerida: depende del dispositivo. 2'99 euros.
InstaFetch	Permite guardar páginas web en el dispositivo para posterior lectura.	Disponible en Google Play. Android 2.0 y superior. Precio: 1.99 euros.
WikiDroyd	Lector *offline* de la *Wikipedia*.	Disponible en Google Play. Android 1.6 y superior. Gratuita.
Note Everything	Permite tomar notas.	Disponible en Google Play. Android 1.5 y superior. Gratuita.

Aplicación	Descripción y funcionalidades	Disponibilidad y Precio
Good Morning	Reloj de alarma que habla y puede funcionar como asistente personal.	Disponible en Google Play. Android 2.2 y superior. Gratuita.
Talking Calendar	Reproduce en voz alta eventos del calendario.	Disponible en Google Play. Android 1.5 y superior. 2.48 euros.
ElkDroid	Automatización del hogar.	Disponible en Google Play. Android 1.5 y superior. 7,73 euros.
Tasker	Automatización total del dispositivo.	Disponible en Google Play. Version requerida: depende del dispositivo. 2'99 euros.
Moon+ Reader Pro	Lector de *ebooks*.	Disponible en Google Play. Android 2.2 y superior. 3,85 euros.
Biblia	Lector de biblia.	Disponible en Google Play. Android 1.6 y superior. Gratuita.
Google Maps	Permite llega rápidamente a cualquier lugar con las indicaciones GPS detalladas por voz. Necesita Text-To-Speech Extended (librería de síntesis de texto a voz que extiende la funcionalidad del la *TTS* API de Android).	Viene instalada por defecto en la mayoría de dispositivos. Gratuita. La librería Text-To-Speech Extended se descarga de forma gratuita en Google Play.
CueBrain	Permite aprender idiomas de manera interactiva.	Disponible en Google Play. Android 1.6 y superior. Gratuita.
DriveCarefully	Permite leer los mensajes de texto y el email durante la conducción.	Disponible en Google Play. Android 2.1 y superior. Gratuita.
Traductor	Traductor que incorpora más de 50 idiomas.	Disponible en Google Play. Version requerida: depende del dispositivo. Gratuita.
Talk To Me Classic	Traductor de entrada y salida por voz.	Disponible en Google Play. Android 1.6 y superior. Gratuita.

Aplicación	Descripción y funcionalidades	Disponibilidad y Precio
Bike Hub Cycle Journey planner	Busca la ruta más rápida para el recorrido en bicicleta.	Disponible en Google Play. Android 1.6 y superior. Gratuita.
SportsTracker Pro	Guarda el recorrido que se realiza. Monitoriza el rendimiento y el pulso cardiaco.	Disponible en Google Play. Version requerida: depende del dispositivo. 3,49 euros.
Intersection Explorer	Ayuda a las personas invidentes a desplazarse por su vecindario ya que dicta en voz alta el mapa de las calles e intersecciones al tocar el mapa y desplazarse por el mediante el tacto.	Disponible en Google Play. Version requerida: depende del dispositivo. Gratuita.

Tabla 2.12: Aplicaciones para dispositivos móviles Android que incorporan TTS

2.5. Conclusiones

A lo largo de este estudio previo, se han visto las posibilidades que ofrece Android para desarrollar aplicaciones que permitan la interacción oral con el usuario, de forma que el usuario pueda interactuar con la aplicación sin la necesidad de leer la pantalla del dispositivo.

En cuanto al reconocimiento de voz, se ha visto que el paquete android.speech del SDK de Android ofrece como posibilidades la integración en una aplicación de servicios de reconocimiento de voz ya existentes o el diseño de servicios de reconocimiento de voz completamente nuevos. La opción más simple consiste en utilizar cualquier servicio de reconocimiento de voz que el dispositivo haya registrado para recibir un *RecognizerIntent* siendo únicamente necesario utilizar las constantes de la clase android.speech.RecognizerIntent. La aplicación de búsqueda por voz de Google, o *Google Search* en inglés, instalada por defecto en la mayoría de los dispositivos Android, responde a un *RecognizerIntent* mostrando una ventana con el texto "Habla ahora". Posteriormente, transfiere la voz emitida por el usuario a los servidores de Google (los mismos que se utilizan cuando el usuario pulsa el botón del micrófono en el *widget* del buscador de Google o en el teclado con entrada de voz habilitada) que se encargan de realizar el reconocimiento. Por lo tanto, en lugar de diseñar un servicio de voz nuevo, resulta más sencillo usar el servicio de voz que ofrece Google mediante su aplicación de búsqueda por voz. La mayor parte de asistentes de voz existentes utilizan dicha opción, ya que el servicio de voz ofrecido por Google ofrece altas prestaciones además de tener una fácil integración en las aplicaciones. Por consiguiente, para diseñar la aplicación multimodal para dispositivos móviles Android que integra reconocimiento de voz como entrada al sistema, se ha utilizado la clase android.speech.RecognizerIntent para enviar un *RecognizerIntent* a la aplicación de búsqueda por voz de Google, siendo necesaria su previa instalación en el dispositivo. Cabe destacar que para utilizar el servicio de reconocimiento de voz de Google es necesario tener acceso a Internet.

En cuanto a la síntesis de texto a voz, el sistema operativo Android 1.6 o superior incorpora en la mayoría de sus dispositivos un motor de síntesis de voz denominado Pico TTS, el cual permite integrar de forma sencilla la síntesis de voz a cualquier aplicación mediante la utilización del paquete android.speech.tts, en particular mediante la utilización de la clase TextToSpeech perteneciente a dicho paquete. No obstante, como se ha visto en este estudio, Android ofrece la posibilidad de instalar y personalizar varios motores además del motor Pico TTS, aunque siempre eligiendo uno como motor TTS principal. Para seleccionar el motor de síntesis de voz utilizado en la aplicación multimodal para dispositivos móviles Android desarrollada, se ha tenido en cuenta la calidad de las voces que ofrece,

los idiomas que tiene disponibles para Android, el tamaño de la descarga, la disponibilidad y el precio. En base al estudio de varios motores de síntesis de voz existentes, se ha decidido emplear el motor de síntesis IVONA TTS HQ, ya que ofrece multitud de voces femeninas y masculinas cuya calidad es la mejor junto a la de las voces del motor SVOX Classic TTS, pero a diferencia de este último, el motor IVONA TTS HQ es actualmente gratuito. Además, se encuentra disponible en *Google Play* por lo que se puede descargar de forma sencilla.

Capítulo 3

DESCRIPCIÓN GENERAL DEL SISTEMA

En este capítulo se realiza una descripción general de la aplicación para dispositivos móviles Android y de las tecnologías empleadas para su desarrollo en el presente Proyecto Final de Carrera.

En primer lugar, se realiza una presentación general del sistema. Para ello, se presenta la arquitectura modular del sistema y se realiza una descripción general de cada uno de sus módulos en cuanto a funcionalidad, arquitectura y tecnologías utilizadas.

Finalmente, se realiza un análisis de las tecnologías empleadas en desarrollo del sistema y se describe la implementación de las operaciones generales.

3.1. Presentación del sistema

La aplicación para dispositivos móviles Android desarrollada en el presente Proyecto Final de Carrera consiste en un asistente virtual multimodal cuya función es la de proporcionar, a través de un sistema de diálogo multimodal, información televisiva y de cine atendiendo a los gustos registrados por el usuario.

El esquema general del sistema diseñado se muestra en la Figura 3.1. Al ser una aplicación multimodal, la aplicación permite al usuario introducir su petición mediante el teclado o mediante la voz. Si se realiza la consulta mediante la voz, el módulo de reconocimiento automático del habla procesa la señal de voz emitida por el usuario y reconoce la información contenida en ésta, convirtiéndola en formato texto, siendo este formato el utilizado como entrada a la aplicación. La respuesta textual generada como salida se muestra por pantalla en formato texto o es convertida de texto a voz mediante el módulo de síntesis de texto a voz.

Figura 3.1: Arquitectura modular del sístema multimodal diseñado

3.1.1. Implementación de los módulos de Reconocimiento Automático del Habla y de Síntesis de Texto a Voz

En el Capítulo 2, se ha realizado un estudio de las posibilidades que ofrece Android para el desarrollo de aplicaciones que permitan la interacción oral con el usuario, de forma que el usuario pueda interactuar con la aplicación sin la necesidad de escribir por teclado ni de leer la pantalla del dispositivo. A continuación, se realiza una explicación de las opciones seleccionadas para la implementación de los módulos de Reconocimiento Automático del Habla (RAH/ASR) y de Síntesis de Texto a Voz (TTS).

Reconocimiento Automático del Habla (RAH /ASR)

La opción más simple consiste en utilizar cualquier servicio de reconocimiento de voz que el dispositivo haya registrado para recibir un *RecognizerIntent* siendo únicamente necesario utilizar las constantes de la clase android.speech.RecognizerIntent del SDK de Android. La aplicación de búsqueda por voz de Google, instalada por defecto en la mayoría de los dispositivos Android, responde a un *RecognizerIntent* mostrando una ventana con el texto "Habla ahora". Posteriormente, transfiere la voz emitida por el usuario a los servidores de Google que se encargan de realizar el reconocimiento. Finalmente, los resultados del reconocimiento de voz son devueltos a la aplicación para que puedan ser procesados.

La mayor parte de asistentes de voz existentes utilizan el servicio de voz ofrecido por Google ya que ofrece altas prestaciones además de tener una fácil integración en las aplicaciones. Por consiguiente, para integrar el reconocimiento de voz en la aplicación para dispositivos móviles Android, se ha utilizado la clase android.speech.RecognizerIntent del SDK de Android con el objetivo de enviar un *RecognizerIntent* a la aplicación de búsqueda por voz de Google, siendo necesaria su previa instalación en el dispositivo. Cabe destacar que para utilizar el servicio de reconocimiento de voz de Google es necesario tener acceso a Internet ya que muchos dispositivos no soportan el reconocimiento de voz *offline*.

Síntesis de Texto a Voz (TTS)

El sistema operativo Android 1.6 o superior incorpora en la mayoría de sus dispositivos un motor de síntesis de voz denominado Pico TTS o Google TTS, el cual permite integrar de forma sencilla la síntesis de voz a cualquier aplicación mediante la utilización del paquete android.speech.tts perteneciente al SDK de Android, en particular mediante la utilización de su clase android.speech.tts.TextToSpeech. No obstante, como se ha visto en el Capítulo 2, Android ofrece la posibilidad de instalar y personalizar varios motores, aunque siempre eligiendo uno como motor de síntesis principal.

A la hora de elegir el motor de síntesis utilizado en la aplicación se ha tenido en cuenta la calidad de las voces que ofrece, los idiomas que tiene disponibles para Android, el tamaño de la descarga, la disponibilidad y el precio. Tras el estudio de los motores existentes, se ha decidido emplear el motor de síntesis IVONA TTS HQ, ya que ofrece multitud de voces femeninas y masculinas cuya calidad es la mejor junto a la de las voces del motor SVOX Classic TTS, pero a diferencia de este último, el motor IVONA TTS HQ es actualmente gratuito. Además, se encuentra disponible en Google Play por lo que se puede descargar de forma sencilla.

3.1.2. Aplicación multimodal desarrollada para dispositivos móviles Android: Asistente Televisivo y de Cine

La aplicación para dispositivos móviles Android desarrollada en el presente Proyecto Final de Carrera se implementa mediante la arquitectura modular ilustrada en la Figura 3.2 y a continuación, se presentan los aspectos más importantes en cuanto a funcionalidad e implementación de los módulos que forman parte de dicha arquitectura. No obstante, dichos módulos se explican de forma detallada en el Capítulo 4.

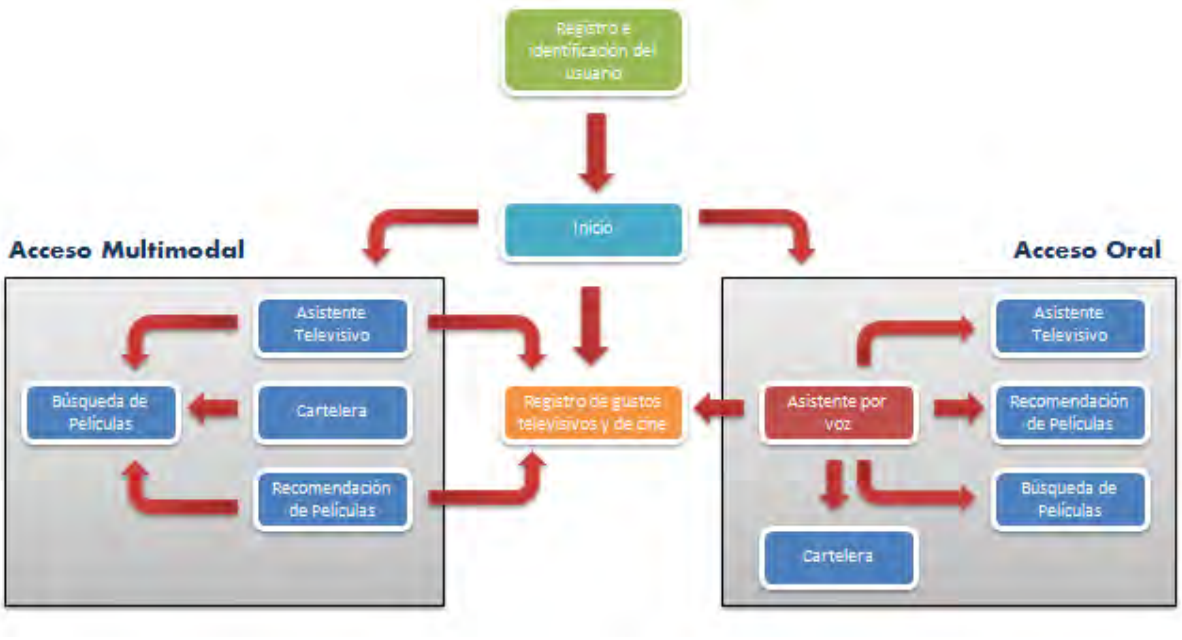


Figura 3.2: Arquitectura modular de la aplicación multimodal desarrollada para dispositivos móviles Android

Módulo Registro e identificación del usuario

En su primera interacción con la aplicación Android, el usuario registra el nombre de usuario y contraseña. Además, rellena un formulario con sus gustos televisivos y de cine. Posteriormente, la aplicación Android (cliente) envía dicha información a un servidor web que accede a una base de datos MySQL remota con el objetivo de almacenar los datos de usuario en una tabla de usuarios registrados. Por consiguiente, la tabla de usuarios registrados en la base de datos MySQL almacena la siguiente información de usuario:

- El nombre de usuario y contraseña de los usuarios registrados.
- Las preferencias del usuario en cuanto a la programación televisiva (deportes, cine, documentales, música, noticias, series TV, programas, *realities* y concursos).
- Las preferencias sobre cine del usuario en cuanto a género, país, rango de años entre los cuales se desea obtener resultados, y a si se desea excluir series de televisión o documentales de los resultados.

En posteriores interacciones con el sistema, el usuario podrá iniciar sesión si se identifica a través del nombre de usuario y contraseña registrados. Una vez que el usuario ha sido identificado correctamente, la información sobre sus gustos televisivos y de cine almacenada en la tabla de usuarios registrados de la base de datos MySQL, es enviada a la aplicación móvil para que esta la almacene en una base de datos interna SQlite. De esta forma, el usuario puede identificarse desde cualquier dispositivo Android y tener acceso a sus gustos televisivos y de cine sin la necesidad de volver a rellenar el formulario. No obstante, la aplicación Android dispone también de un modo *offline* que permite al usuario acceder a los servicios sin la necesidad de identificarse.

El registro e identificación de usuario se ha desarrollado utilizando la arquitectura cliente-servidor que se muestra en la Figura 3.3. Tal y como se observa, para la implementación de la parte web se ha utilizado como lenguaje en la parte de servidor PHP (Versión 5.4), como base de datos MySQL y como servidor web Apache. La aplicación Android (cliente) y el servidor se comunican mediante la utilización de JSON y HTTP. La aplicación Android accede a la base de datos interna SQLite para almacenar la información sobre los gustos televisivos y de cine.

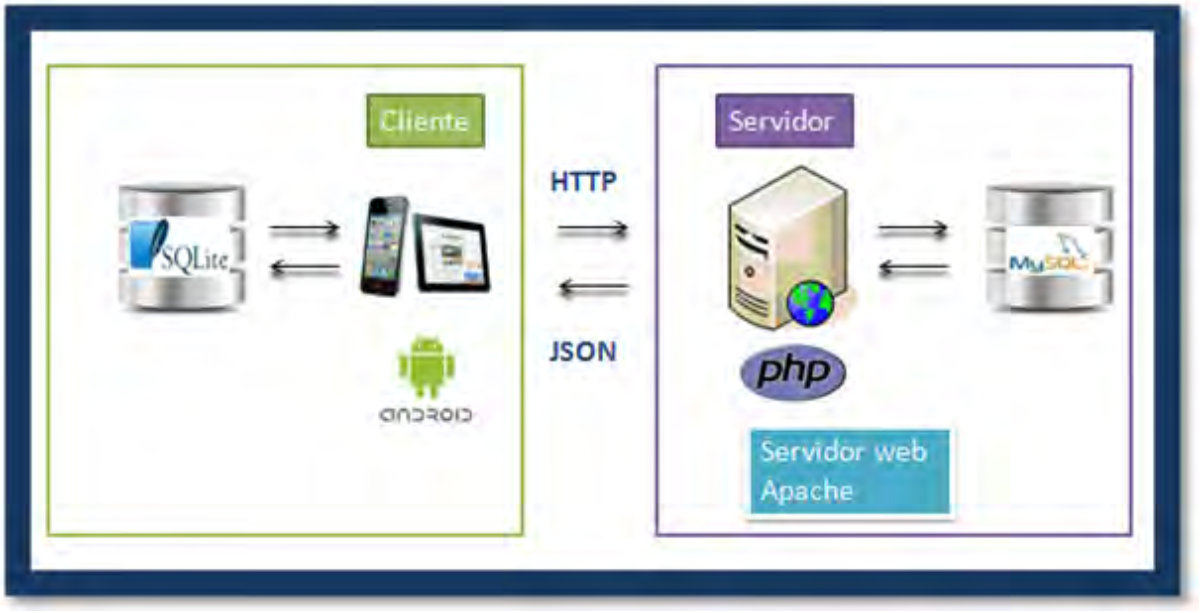


Figura 3.3: Arquitectura del módulo de registro e identificación del usuario

Módulo Inicio

Este módulo muestra un menú cuyas opciones permiten al usuario dirigirse a los módulos ***Registro de gustos televisivos y de cine***, ***Asistente televisivo***, ***Recomendación de películas***, ***Búsqueda de películas***, ***Cartelera*** y ***Asistente de voz***. Además, permite al usuario cerrar sesión y dirigirse de nuevo al módulo ***Registro e identificación del usuario***.

Módulo Registro de gustos televisivos y de cine

Para poder utilizar los servicios ofrecidos por los módulos ***Asistente televisivo*** y ***Recomendación de películas***, es necesario que el usuario registre sus preferencias en cuanto a la programación televisiva y de cine en un formulario.

El registro de gustos televisivos y de cine se realiza por primera vez durante el registro del usuario en el módulo ***Registro e identificación del usuario***. Tal y como se ha explicado anteriormente, una vez completado el formulario por el usuario, la aplicación Android envía dicha información a la base de datos MySQL remota, rellenando la tabla de usuarios registrados para el usuario correspondiente. Posteriormente, cuando el usuario se identifica en el sistema correctamente, la información sobre sus gustos televisivos y de cine almacenada en la tabla de usuarios registrados de la base de datos MySQL, es enviada a la aplicación móvil para que esta la almacene en una base de datos interna SQlite.

La aplicación Android permite al usuario volver a rellenar el formulario siempre que lo desee. Una vez que el usuario modifique sus gustos a través del formulario, se envía dicha información a la base de datos MySQL remota con el objetivo de actualizar la tabla de usuarios registrados para el usuario correspondiente. Además, se envía también dicha información a la base de datos SQLite.

Tal y como se ha explicado anteriormente, la aplicación dispone de un modo *offline* en el que no es necesaria la identificación del usuario. En este caso, el usuario puede rellenar el formulario con sus gustos televisivos y de cine pero dicha información se envía únicamente a la base de datos interna SQLite.

El módulo ***Registro de gustos televisivos y de cine*** rellena las siguientes tablas de la base de datos SQLite:

- **Tabla de gustos televisivos**: Almacena las preferencias de un usuario determinado en cuanto a la programación televisiva (deportes, cine, documentales, música, noticias, series TV, programas,

realities y concursos). Dicha tabla es accedida desde el módulo ***Asistente televisivo***.

- **Tabla de gustos de cine**: Almacena las preferencias sobre cine de un usuario determinado en cuanto a género, país, rango de años entre los cuales se desea obtener resultados, y a si se desea excluir series de televisión o documentales de los resultados.

- **Tabla de películas recomendadas al usuario**: Almacena la lista de películas recomendadas al usuario, extraída del contenido HTML de una página web mediante la utilización de la librería de Java JSoup, de acuerdo a los gustos del usuario registrados en la tabla de gustos de cine. Dicha tabla es accedida desde el módulo ***Recomendación de películas***.

Además, la base de datos SQLite contiene unas tablas que almacenan la programación televisiva de varios canales y que son accedidas, junto a la tabla de gustos televisivos, por el módulo ***Asistente televisivo*** para recomendar programación al usuario:

- **Tablas de programación televisiva**: Almacenan la información sobre la programación televisiva (TVE 1, TVE 2, Antena 3, Cuatro, Tele 5, Sexta, TeleMadrid, Sexta 3 y Paramount Channel) extraída del contenido HTML de una página web mediante la utilización de la librería de Java JSoup. Se crea una tabla por cada canal de televisión. La aplicación Android permite al usuario actualizar dicha tabla con la programación del día de la consulta.

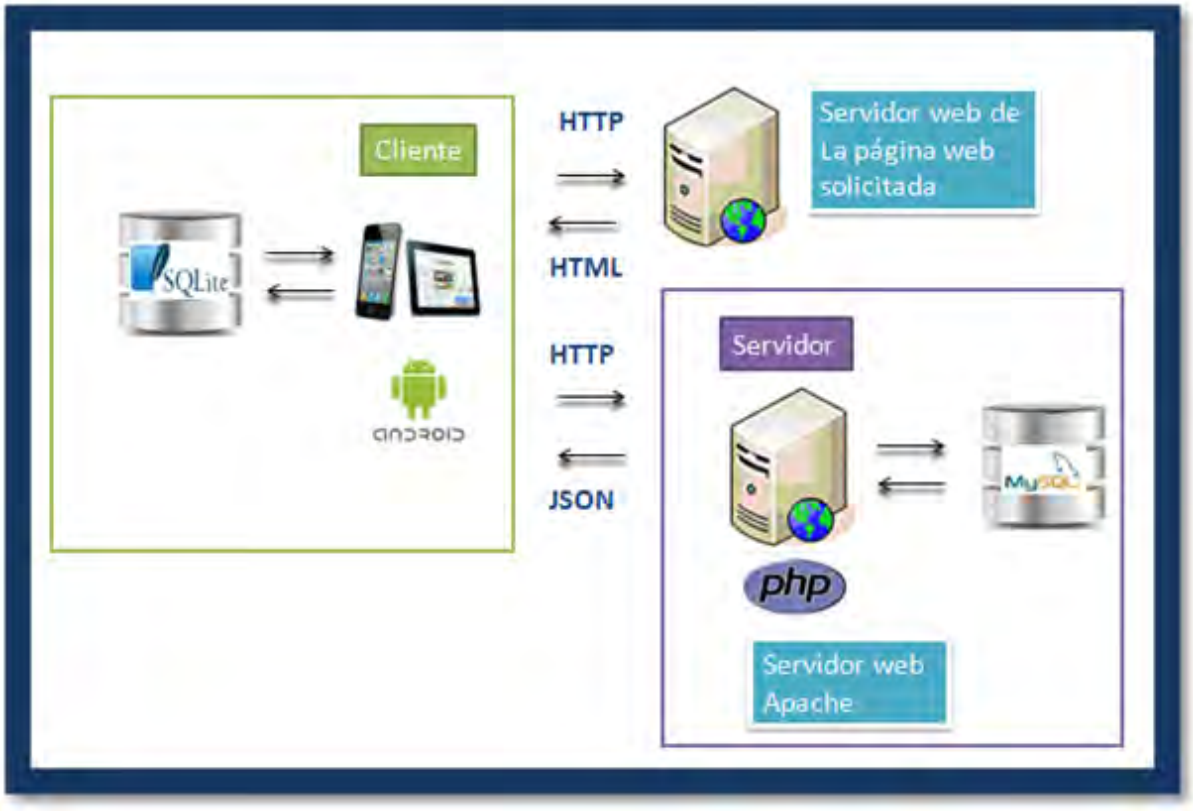


Figura 3.4: Arquitectura del módulo de registro de gustos televisivos y de cine

La Figura 3.4 muestra la arquitectura del módulo de registro de gustos televisivos y de cine. Tal y como se observa, el acceso a la base de datos SQLite se realiza desde el lado del cliente. Para obtener la información a almacenar por las tablas de programación televisiva y de películas recomendadas, la aplicación Android (cliente) se conecta mediante HTTP al servidor web de la página web que contiene dicha información y la extrae del contenido HTML de la página web mediante la utilización de la

librería de Java JSoup. Además, la aplicación Android envía la información sobre los gustos televisivos y de cine al servidor PHP con el objetivo de actualizar la tabla de usuarios registrados de la base de datos MySQL remota.

Servicios ofrecidos al usuario mediante interacción multimodal

La aplicación para dispositivos móviles Android desarrollada en el presente Proyecto Final de Carrera ofrece al usuario el acceso, mediante una interacción multimodal, a cuatro servicios implementados en módulos separados.

- Módulo ***Asistente televisivo***: Recomienda al usuario la programación diaria de acuerdo con los gustos registrados. Para ello, consulta la información almacenada en las tablas de gustos televisivos y de programación televisiva de la base de datos SQLite. Además, el usuario puede consultar información más detallada de cada película, serie de TV y documental recomendados por el asistente, accediendo al módulo ***Búsqueda de películas***. El servicio de asistente televisivo funciona en modo *offline* siendo únicamente necesario disponer de acceso Internet cuando se vaya a actualizar las tablas de programación televisiva de la base de datos SQLite con la programación del día de la consulta. Es necesario haber registrado los gustos televisivos para poder utilizar el presente servicio.

- Módulo ***Cartelera***: Permite al usuario buscar salas de cine y obtener información de la cartelera asociada a dicho cine. Dicha información se extrae del contenido HTML de una página web mediante la utilización de la librería de Java JSoup. Es necesario disponer de acceso a Internet para utilizar este servicio. Además, el usuario puede consultar información más detallada de cada película en cartelera, accediendo al módulo ***Búsqueda de películas***.

- Módulo ***Recomendación de películas***: Recomienda al usuario películas de acuerdo con los gustos registrados. Para ello, consulta la información almacenada en la tabla de recomendación de películas de la base de datos SQLite. Además, el usuario puede consultar información más detallada de cada película, serie de TV y documental recomendados por este módulo, accediendo al módulo ***Búsqueda de películas***. El servicio de recomendación de películas funciona en modo *offline.* Es necesario haber registrado los gustos de cine para utilizar este servicio.

- Módulo ***Búsqueda de películas***: Permite al usuario solicitar información acerca de una película. Dicha información se extrae del contenido HTML de una página web mediante la utilización de la librería de Java JSoup. Es necesario disponer de acceso a Internet para utilizar este servicio.

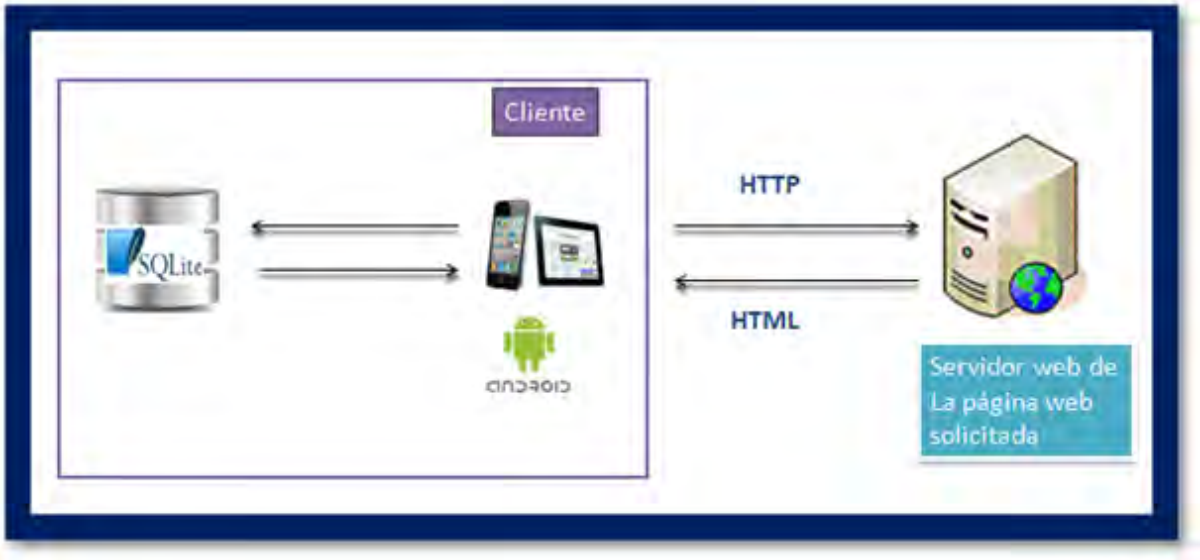
El usuario puede interactuar con el sistema para acceder a estos cuatro servicios mediante el modo táctil, el modo oral, o combinar ambos modos.

La arquitectura del módulo ***Asistente televisivo*** se observa en la Figura 3.5.a. Este módulo necesita acceder a la tabla de gustos televisivos de la base de datos SQLite para poder ofrecer el servicio correspondiente al usuario. Además, para poder actualizar las tablas de programación televisiva de la base de datos SQLite con la programación del día de la consulta, la aplicación Android (cliente) se conecta mediante HTTP al servidor web de la página web que contiene dicha información solicitada por el usuario y la extrae del contenido HTML de la página web mediante la utilización de la librería de Java JSoup. La actualización de las tablas de programación televisiva de la base de datos SQLite se necesita realizar solo una vez al día, por lo que este módulo funciona usualmente en modo *offline*.

La arquitectura del módulo ***Recomendación de películas*** se observa en la Figura 3.5.b. Este módulo necesita acceder a la tabla de películas recomendadas de la base de datos SQLite para poder ofrecer el servicio correspondiente al usuario. Tal y como se observa, el acceso a la base de datos SQLite se realiza desde el lado del cliente por lo que no es necesario disponer acceso a Internet para poder utilizar el servicio ofrecido por este módulo.

La arquitectura del módulo de ***Cartelera*** y del módulo ***Búsqueda de películas*** se observa en la Figura 3.5.c. En dichos módulos, la aplicación Android (cliente) se conecta mediante HTTP al servidor web de la página web que contiene la información solicitada por el usuario y la extrae del contenido HTML de la página web mediante la utilización de la librería de Java JSoup.

A pesar de que los módulos ***Recomendación de películas*** y ***Asistente televisivo*** ofrecen la mayor parte de su funcionalidad sin la necesidad de tener acceso a Internet, hay que tener en cuenta que para utilizar el reconocimiento de voz es necesario disponer de Internet ya que gran parte de los dispositivos no soportan el reconocimiento de voz *offline*, por lo que dichos servicios únicamente podrán utilizarse *offline* a través de los interfaces tradicionales como la pantalla o el teclado.



(a) Arquitectura del módulo Asistente televisivo

(b) Arquitectura del módulo Recomendación de películas

(c) Arquitectura de los módulos Cartelera y Búsqueda de películas

Figura 3.5: Arquitectura de los módulos que implementan los servicios ofrecidos al usuario

Servicios ofrecidos al usuario mediante interacción oral

El módulo ***Asistente por voz*** conduce al usuario a los a los módulos ***Asistente televisivo***, ***Recomendación de películas***, ***Búsqueda de películas***, y ***Cartelera*** en función de las decisiones que vaya tomando en cada diálogo. Este módulo utiliza únicamente el reconocedor de voz como entrada al sistema y la síntesis de texto a voz como salida al sistema.

El servicio ofrecido por los cuatro módulos es idéntico al ofrecido en el caso multimodal a excepción de los siguientes puntos:

- Es necesario disponer de acceso a Internet para acceder a los cuatro módulos y utilizar sus servicios ya que el reconocimiento de voz únicamente funciona en la mayoría de dispositivos

Android si se dispone de acceso a internet y es la única opción de entrada al sistema en el modo oral.

- El modo multimodal no permitía al usuario seleccionar mediante la voz la opción deseada de la lista de resultados de las películas encontradas (modulo ***Búsqueda de películas***) y de la lista con salas de cine encontradas (módulo ***Cartelera***). El modo oral si lo permite, ya que se asocia a cada elemento de la lista de resultados a un número y basta con que el usuario diga el número para seleccionar el elemento de la lista sobre el que desea obtener información.
- En el modo multimodal, el usuario podía consultar información más específica acerca de las películas, documentales o series de televisión recomendadas en los módulos ***Asistente televisivo***, ***Recomendación de películas***, y ***Cartelera.*** Esta función aún no está disponible para el módulo ***Asistente por voz*** del menú principal que permite al usuario interactuar con la aplicación únicamente de forma oral.

La arquitectura para los módulos ***Asistente televisivo***, ***Recomendación de películas***, ***Búsqueda de películas***, y ***Cartelera*** en modo oral es igual a la del modo multimodal (Figura 3.5).

3.2. Tecnologías utilizadas

En la presente sección se analizan las tecnologías utilizadas en el desarrollo de la aplicación para dispositivos móviles Android. Puesto que la aplicación desarrollada para el presente Proyecto Final de Carrera se ha desarrollado para la plataforma Android, se va a dedicar el primer apartado al estudio de dicha plataforma. Además se explica el proceso que se ha seguido para preparar el entorno de desarrollo de la aplicación.

En el siguiente apartado, se realiza una introducción al sistema de gestión de base de datos SQLite y se describe como crear, gestionar y manipular una base de datos SQLite en Android.

Finalmente, se analizan las tecnologías utilizadas en el módulo *Registro e identificación de usuario* para el cual se ha desarrollado un servicio web con el objetivo de almacenar en una base de datos MySQL remota una tabla de usuarios registrados.

3.2.1. Plataforma Android

Android es una pila de software de código abierto para dispositivos móviles, creada por Google y OHA, que incluye (Nimodia y Deshmukh, 2012):

- Un sistema operativo basado en GNU Linux
- *Middleware*, o software intermedio, para facilitar el desarrollo de aplicaciones
- Aplicaciones comunes ya instaladas para facilitar el uso

Las características principales de Android se listan a continuación:

- El Marco de la aplicación permite la reutilización y sustitución de componentes.
- Máquina virtual adaptada a dispositivos móviles: Maquina virtual Dalvik
- Navegador integrado basado en el motor de código abierto WebKit
- Biblioteca de gráficos 2D y 3D OpenGL ES 1.0
- SQLite para el almacenamiento de datos estructurados
- Soporte para audio, vídeo, y diferentes formatos de imagen
- Telefonía GSM
- Interfaces de red: GSM, CDMA, EDGE, 3G, Bluetooth y Wi-Fi

- Cámara, GPS, brújula, y acelerómetro
- Entorno de desarrollo completo que incluye un emulador de dispositivos, herramientas para la depuración, la memoria y perfiles de rendimiento, y un plug-in para el IDE de Eclipse

3.2.1.1. Arquitectura de la plataforma Android

La arquitectura de Android está formada por varias capas que facilitan al desarrollador la creación de aplicaciones. La Figura 3.6 muestra el diagrama de la arquitectura de Android.

Figura 3.6: Arquitectura de Android

La distribución que se observa en la Figura 3.6 permite acceder a las capas más bajas mediante el uso de librerías y que así el desarrollador no tenga que programar a bajo nivel para que una aplicación acceda a los componentes de hardware de los dispositivos. Cada una de las capas utiliza elementos de la capa inferior para realizar sus funciones, es por ello que a este tipo de arquitectura se le conoce también como pila. En los siguientes apartados se explica cada una de las capas que componen la estructura del sistema operativo de Android (Vico, 2011).

Kernel de Linux

El núcleo del sistema operativo Android está basado en el kernel de Linux versión 2.6. Proporciona una capa de abstracción para los elementos hardware a los que acceden las aplicaciones. Para cada elemento hardware del dispositivo existe un controlador (*driver*) dentro del kernel que permite utilizarlo desde el software.

Además de proporcionar controladores hardware, el kernel se encarga de gestionar los diferentes recursos del teléfono (energía, memoria, etc.) y del sistema operativo en sí: procesos, elementos de comunicación, etc.

Librerías

La capa que se sitúa justo sobre el kernel la componen las librerías de Android (*Libraries*). Estas librerías están escritas en C o C++ y compiladas para la arquitectura hardware específica del teléfono. Dichas librerías se encuentran instaladas en el terminal antes de ser puesto en venta. Su cometido es proporcionar funcionalidad a las aplicaciones, para tareas que se repiten con frecuencia, evitando tener que codificarlas cada vez y garantizando su eficiencia. La Tabla 3.1 resume las librerías que se incluyen normalmente y que forman parte de la presente capa de la arquitectura Android.

Librería	Función
Gestor de superficies (*Surface Manager*)	Se encarga de componer las imágenes que se muestran en la pantalla a partir de capas gráficas 2D y 3D.
Bibliotecas multimedia (*Media Framework*)	Permiten visualizar, reproducir e incluso grabar numerosos formatos de imagen, vídeo y audio como JPG, GIF, PNG, MPEG4, AVC (H.264), MP3, AAC o AMR.
SQLite	Motor de bases de datos relacionales, disponible para todas las aplicaciones.
OpenGL | ES (*OpenGL for Embedded Systems*)	Motor gráfico 3D basado en las APIs (*Application Program Interface*) de OpenGL ES 1.0, 1.1 (desde la versión 1.6 de Android) y 2.0 (desde la versión 2.2 de Android).
FreeType	Permite mostrar fuentes tipográficas, tanto basadas en mapas de bits como vectoriales.
WebKit	Motor web utilizado por el navegador (tanto como aplicación independiente como embebido en otras aplicaciones). Es el mismo motor que utilizan Google Chrome y Safari (el navegador de Apple, tanto en Mac como en el iPhone).
SGL (*Scalable Graphics Library*)	Motor gráfico 2D de Android.
SSL (*Secure Sockets Layer*)	Proporciona seguridad al acceder a Internet por medio de criptografía.
Biblioteca C de sistema (libc)	Proporciona funcionalidad básica para la ejecución de las aplicaciones.

Tabla 3.1: Librerías que forman parte de la arquitectura Android

Entorno de ejecución

El entorno de ejecución de Android (*Android Runtime*) se apoya en las librerías descritas anteriormente aunque no se considera una capa en sí mismo. Dicho entorno de ejecución está formado por las librerías esenciales de Android, que incluyen la mayoría de la funcionalidad de las librerías habituales de Java así como otras específicas de Android.

El componente principal del entorno de ejecución de Android es la máquina virtual Dalvik, componente que ejecuta todas y cada una de las aplicaciones no nativas de Android. Las aplicaciones se codifican normalmente en Java y se compilan en un formato específico para la máquina virtual Dalvik, que se encarga de ejecutarlas. Esto permite compilar una única vez las aplicaciones y distribuirlas ya compiladas teniendo la garantía de que podrán ejecutarse en cualquier dispositivo Android que disponga de la versión mínima del sistema operativo que requiera cada aplicación.

Aunque las aplicaciones se escriben en Java, Dalvik no es compatible con el *bytecode* Java. Java se usa únicamente como lenguaje de programación, pero los ejecutables que se generan con el SDK de Android no son ejecutables Java convencionales. Durante el proceso de compilación de los programas Java (normalmente archivos .java) se genera, de forma intermedia, el *bytecode* habitual (archivos .class). Pero dichos archivos son convertidos al formato específico de Dalvik en el proceso final (.dex, de *Dalvik executable*). La ventaja de generar archivos .dex es que son más compactos que los .class equivalentes (hasta un 50 % menos de tamaño), lo que permite ahorrar espacio en el teléfono y acelerar el proceso de carga. Además, a diferencia de otras máquinas virtuales, Dalvik se basa en registros en lugar de una pila para almacenar los datos, lo que requiere menos instrucciones. Esto permite ejecuciones más

rápidas en un entorno con menos recursos.

Las aplicaciones Android se ejecutan cada una en su propia instancia de la máquina virtual Dalvik, evitando así interferencias entre ellas, y tienen acceso a todas las bibliotecas mencionadas anteriormente y, a través de ellas, al hardware y al resto de recursos gestionados por el kernel.

Marco de aplicación

El marco de la aplicación (*Application Framework*) lo forman todas las clases y servicios utilizados directamente por las aplicaciones para realizar sus funciones y que se apoyan en las librerías y en el entorno de ejecución. La mayoría de los componentes de esta capa son bibliotecas Java que acceden a los recursos a través de la máquina virtual Dalvik. La Tabla 3.2 resume las bibliotecas más importantes de dicha capa de la Arquitectura Android.

Librería	Función
Administrador de actividades (*Activity Manager)*	Permite controlar el ciclo de vida de las actividades y la pila de actividades.
Administrador de ventanas (*Windows Manager*)	Permite organizar lo que se muestra en pantalla, creando superficies que pueden ser rellenadas por las actividades
Proveedor de contenidos (*Content Provider*)	Permite encapsular un conjunto de datos que va a ser compartido entre aplicaciones
Vistas (*Views*	Android proporciona vistas con las que construir las interfaces de usuario: botones, cuadros de texto, listas, etc. También proporciona otras más sofisticadas, como un navegador web o un visor de Google Maps.
Administrador de notificaciones (*Notification Manager*)	Proporciona servicios para notificar al usuario cuando algo requiera su atención.
Administrador de paquetes (*Package Manager*)	Permite obtener información sobre los paquetes actualmente instalados en el dispositivo Android, además de gestionar la instalación de nuevos paquetes.
Administrador de telefonía (*Telephony Manager*)	Proporciona acceso a la pila hardware de telefonía del dispositivo Android.
Administrador de recursos (*Resource Manager*)	Proporciona acceso a todos los elementos propios de una aplicación que se incluyen directamente en el código: cadenas de texto traducidas a diferentes idiomas, imágenes, sonidos, disposiciones de las vistas dentro de una actividad (*layouts*), etc.
Administrador de ubicaciones (*Location Manager*)	Permite determinar la posición geográfica del dispositivo Android (usando el GPS o las redes disponibles) y trabajar con mapas.
Administrador de sensores (*Sensor Manager*)	Permite gestionar todos los sensores hardware disponibles en el dispositivo Android: acelerómetro, giroscopio, sensor de luminosidad, sensor de campo magnético, brújula, sensor de presión, sensor de proximidad, sensor de temperatura, etc.
Cámara	Proporciona acceso a las cámaras del dispositivo Android.
Multimedia	Conjunto de bibliotecas que permiten reproducir y visualizar audio, vídeo e imágenes en el dispositivo.

Tabla 3.2: Biliotecas del Marco de Aplicación de la arquitectura Android

Aplicaciones

La capa superior de la pila software la forman todas las aplicaciones del dispositivo: las que tienen interfaz de usuario y las que no, las nativas (programadas en C o C++), las administradas (programadas en Java), las que vienen por defecto con el dispositivo, las instaladas por el usuario, la aplicación principal del sistema: Inicio (*Home*), etc.

Lo principal a tener en cuenta de esta arquitectura es que todas las aplicaciones utilizan el mismo marco de aplicación para acceder a los servicios que proporciona el sistema operativo. Esto implica que se pueden crear aplicaciones que usen los mismos recursos que usan las aplicaciones nativas y que se puede reemplazar cualquiera de las aplicaciones del teléfono por otra.

3.2.1.2. Componentes de una aplicación en Android

Todas las aplicaciones en Android pueden descomponerse en cinco componentes (Aranaz, 2009). Cada aplicación será una combinación de uno o más de estos componentes, que deberán ser declarados de forma explícita en un fichero con formato XML denominado **AndroidManifest.xml**, junto a otros datos asociados como valores globales, clases que implementa, datos que puede manejar, permisos, etc. En los apartados de a continuación se resumen los cinco componentes en los que puede dividirse una aplicación para dispositivos móviles Android.

Activity

Un componente *Activity* refleja una actividad llevada a cabo por una aplicación, y que lleva asociada típicamente una interfaz de usuario (vistas representadas por subclases View). Este componente se implementa mediante la clase Activity.

La mayoría de las aplicaciones permiten la ejecución de varias acciones a través de la existencia de una o más ventanas. Cada una de estas ventanas debería estar representada a través de un componente *Activity*, de forma que navegar de una ventana a otra implica lanzar una actividad o dormir otra.

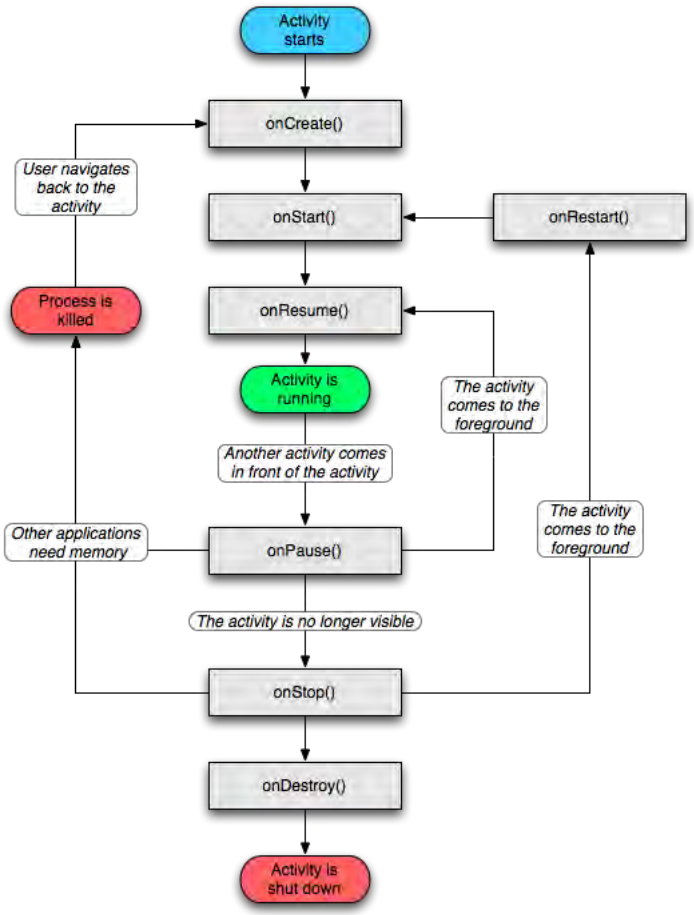


Figura 3.7: Ciclo de vida de un objeto de la clase Activity

La Figura 3.7 muestra el ciclo de vida de un objeto de la clase Activity. Tal y como se observa, un objeto de la clase Activity puede implementar siete métodos diferentes:

- onCreate(), onDestroy(): abarcan todo el ciclo de vida, representan el principio y el fin de la actividad.
- onStart(), onStop(): representan la parte visible del ciclo de vida. Desde *onStart()* hasta *onStop()*, la actividad será visible para el usuario, aunque es posible que no tenga el foco de acción por existir otras actividades superpuestas con las que el usuario está interactuando. Pueden ser llamados múltiples veces.
- onResume(), onPause(): delimitan la parte útil del ciclo de vida. Desde *onResume()* hasta *onPause()*, la actividad no sólo es visible, sino que además tiene el foco de la acción y el usuario puede interactuar con ella.
- onRestart(): despierta una actividad que está en segundo plano o invisible.

Intent

Muy vinculado al componente *Activity* se encuentra el componente *Intent*, que consiste en la voluntad de realizar alguna acción, generalmente asociada a unos datos. Lanzando un *Intent*, una aplicación puede delegar el trabajo en otra (Ej. abrir una URL en algún navegador web), de forma que el sistema se encarga de buscar qué aplicación entre las instaladas es la que puede llevar a cabo la acción solicitada. Este componente se implementa a través de la clase Intent.

Broadcast Intent Receiver

El componente *Broadcast Intent Receiver* se utiliza para lanzar alguna ejecución dentro de la aplicación actual cuando un determinado evento se produce (generalmente, abrir un componente Activity). Este componente se implementa a través de la clase BroadcastReceiver.

Service

EL componente *Service* representa una aplicación ejecutada sin interfaz de usuario, generalmente en segundo plano, mientras otras aplicaciones (con interfaz) están activas en la pantalla del dispositivo. Este componente se implementa a través de la clase Service.

Content Providers

El componente Content Provider es un gestor de contenidos que permite a cualquier aplicación en Android almacenar datos en un fichero, en una base de datos SQLite o en cualquier otro formato. Una clase que implemente dicho componente contendrá una serie de métodos que permiten almacenar, recuperar, actualizar y compartir datos entre distintas aplicaciones. Existe una colección de clases para distintos tipos de gestión de datos en el paquete android.provider.

3.2.1.3. Preparación del entorno de desarrollo para aplicaciones Android en Windows 7

El proceso de preparación del entorno de desarrollo para aplicaciones Android que se ha utilizado consta de los pasos que se enumeran a continuación.

1. Instalar JDK (*Java Development Kit*): kit de desarrollo de Java.
2. Instalar Eclipse: herramienta para desarrollar software.
3. Descargar el SDK (*Software Development Kit*) de Android: kit de desarrollo de aplicaciones Android.

4. Instalar el plug-in ADT (*Android Development Tools*) para Eclipse: para poder desarrollar aplicaciones Android utilizando Eclipse.

5. Agregar plataformas y componentes al SDK: descarga del resto de componentes necesarios para desarrollar aplicaciones.

6. Configurar el plug-in de Eclipse para Android.

La página web de descarga del SDK de Android (`http://developer.android.com/sdk/index.html`) ofrece al desarrollador la opción de descargarse el paquete *SDK ADT Bundle* para Windows. Dicho paquete incorpora Eclipse, el plug-in ADT, el SDK de Android, y todas las herramientas y componentes necesarios para el desarrollo de aplicaciones. Sin embargo, si se desea trabajar con una versión concreta de Eclipse se realizan los pasos de preparación del entorno de desarrollo por separado. A continuación se describe cada paso del proceso de preparación del entorno de desarrollo.

Instalación de Java Development Kit en Windows 7

1. Se descarga la versión actual (JDK 7) desde la página de Java (`http://www.oracle.com/technetwork/java/javase/downloads/index.html`). Una vez descargado dicho archivo, se hace doble clic para que empiece la instalación.

2. Al terminar la instalación del Java JDK, se pide la carpeta de destino donde se tiene que instalar el JRE (*Java Runtime Environment*) que es necesario para ejecutar los programas en Java. Se deja la ruta que viene por defecto y se comienza la instalación.

3. Una vez terminada la instalación de Java JDK y Java JRE, se configuran las variables de entorno. Para ello, se va a la ruta Panel de control->Sistema y seguridad->Sistema y se selecciona la opción "Configuración Avanzada del Sistema", que se encuentra a la izquierda. La Figura 3.8 muestra la ventana que se muestra al usuario cuando se selecciona dicha opción.

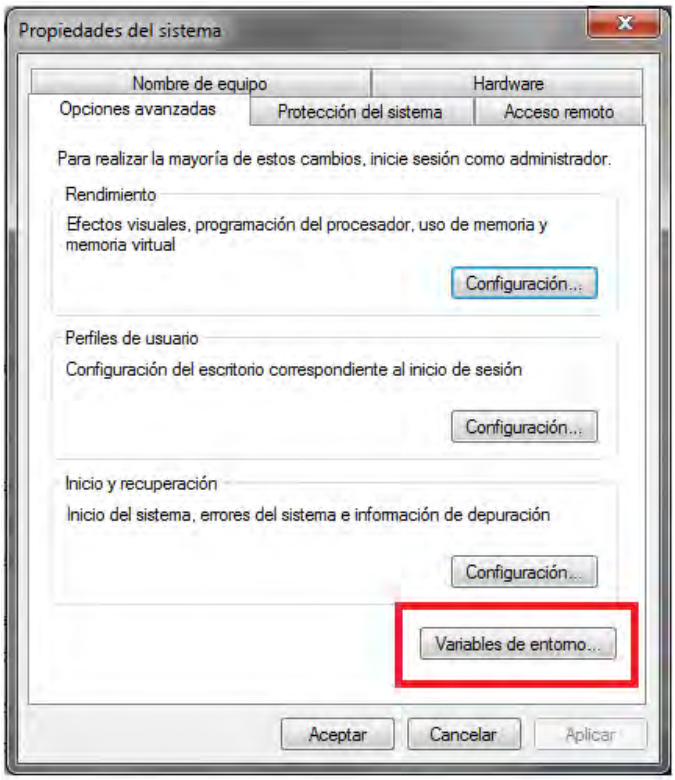


Figura 3.8: Ventana de propiedades del sistema en Windows 7

Se hace clic en "Variables de Entorno" de la Figura 3.8 y se muestra la ventana de la Figura 3.9. En "Variables del sistema", se busca la variable llamada "Path" y se edita para agregar la ruta donde se encuentra instalado el JDK. Posteriormente, se crea una nueva variable llamada "Classpath" con la dirección del fichero src.zip.

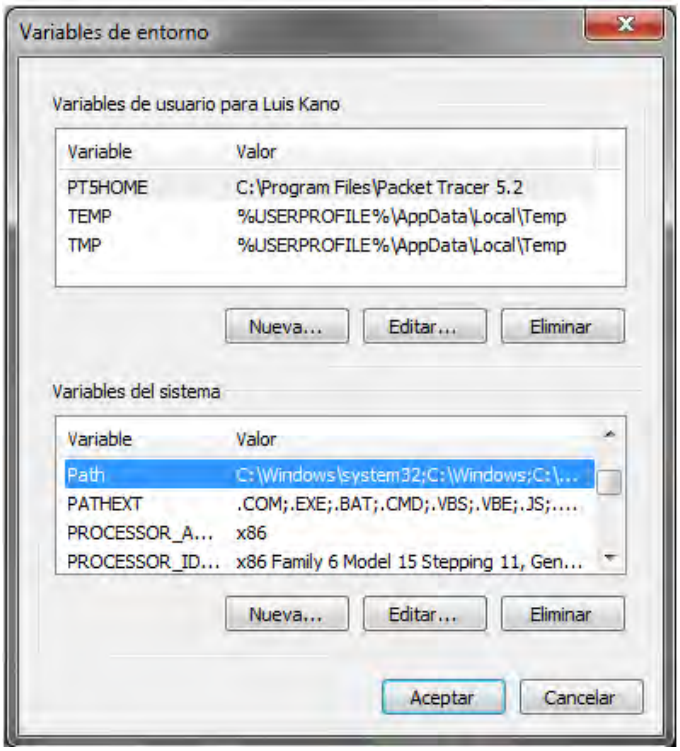


Figura 3.9: Variables de Entorno en Windows 7

Instalación de Eclipse

La descarga de eclipse se realiza desde su página web (`http://www.eclipse.org/downloads/`). Se debe descargar la versión 3.6.2 (Helios) o superior ya que la versión 3.5 (Galileo) no es comptalible con la última version del plug-in ADT. Para el desarrollo del presente Proyecto Fin de Carrera, se ha descargado la versión Eclipse Classic 4.2.1 (Juno) compatible con el sistema operativo Windows. Una vez se ha realizado la descarga, se descomprime el paquete en una carpeta y se ejecuta eclipse haciendo doble click en el ejecutable de dicha carpeta.

Descarga del SDK de Android

Para la descarga del SDK de Android, el desarrollador debe abrir a la página web del SDK de Android (`http://developer.android.com/sdk/index.html`) y descargarse el SDK comprimido: android-sdk_r22.3-windows.zip. Posteriormente, se descomprime en una carpeta.

Instalación del plug-in ADT

Para instalar el plug-in ADT en eclipse, se debe abrir Eclipse e ir al menú *Help->Install New Software*. El menú de instalación se observa en la Figura 3.10. Se debe indicar la dirección del repositorio en el que se encuentra el software que se va a instalar. La dirección del repositiorio es: `https://dl-ssl.google.com/android/eclipse/`. Dicha dirección hay que copiarla en el campo *work with* y añadirla (*add*). Cuando pregunte el nombre, se puede dar aquél que se considere oportuno. Pos-

teriormente, se selecciona todas las funcionalidades y se pulsa *Next*. Finalmente se acepta el acuerdo. Cuando la instalación termine, se debe reiniciar Eclipse.

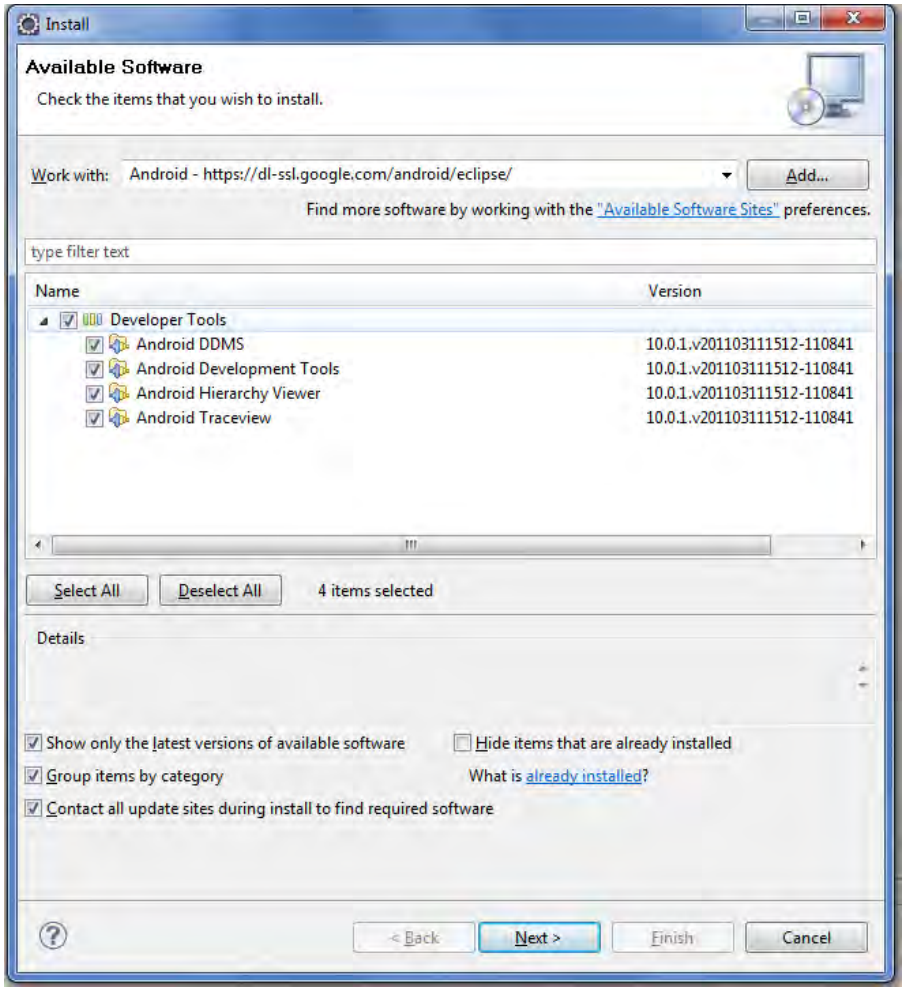


Figura 3.10: Ventana de instalación del plug-in ADT de Eclipse

Agregar plataformas y componentes al SDK

Para descargar la última versión de las herramientas y las plataformas del SDK mediante el Gestor del SDK, se debe ir a la carpeta dónde está descomprimido el SDK de Android y hacer doble click sobre SDKManager.exe. Una vez en el Gestor del SDK, se selecciona la opción *Accept All* y se inicia la instalación (*Install*).

Configurar el plug-in de Eclipse para Android

Una vez instalado el plug-in ADT, es necesario indicarle dónde se encuentra el SDK de android. Para ello, se abre el menú *windows>Preferences* y se selecciona la opción *Android* en el panel de la izquierda, tal y como se observa en la Figura 3.11. Se hace click en *Browse* y se selecciona la carpeta en la que se encuentra el SDK de Android que se ha descargado.

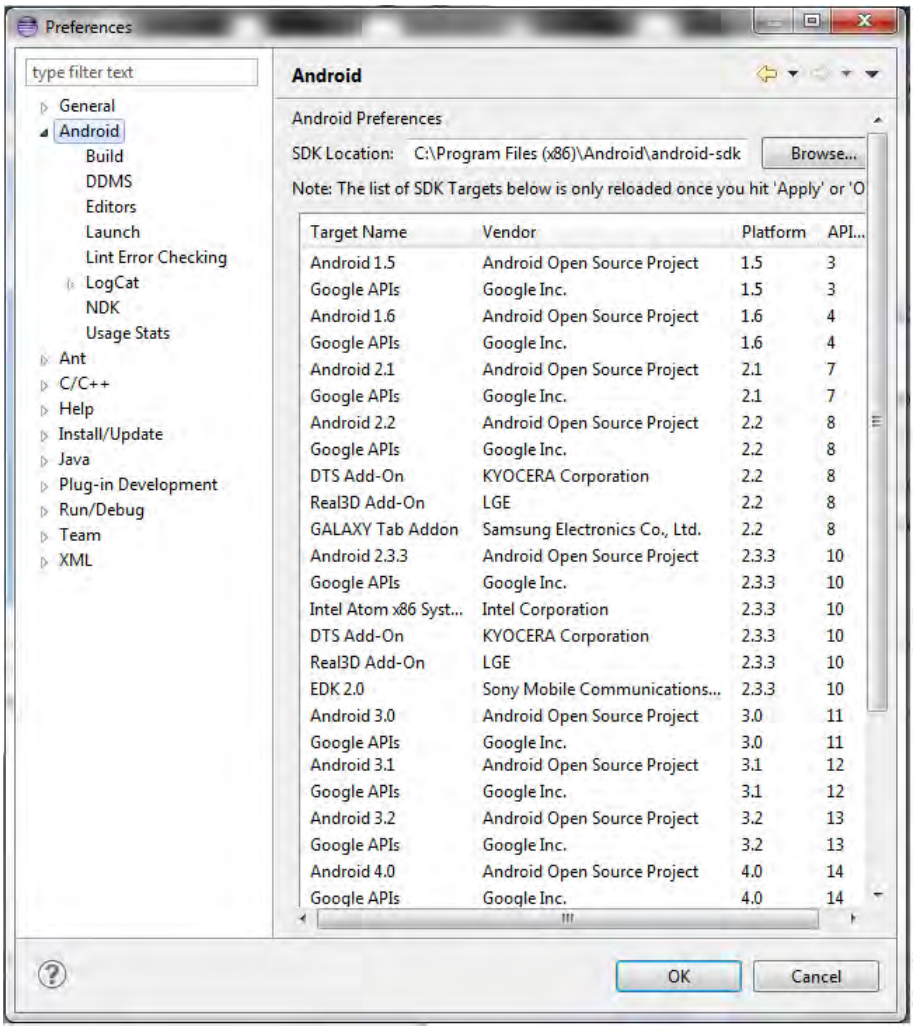


Figura 3.11: Ventana de configuración del plug-in de Eclipse para Android

3.2.1.4. Desarrollo de una aplicación Android en Eclipse

Creación de un proyecto Android en Eclipse

Las herramientas del SDK facilitan crear un nuevo proyecto Android en Eclipse con un conjunto de ficheros y carpetas por defecto. En primer lugar se ha de seleccionar la opción del menú *File->New->Project->Android Application Project*.

La Figura 3.12 muestra el formulario que se ha de rellenar para la creación de un nuevo proyecto Android. Tal y como se observa, el formulario consta de los campos descritos a continuación:

- **Nombre de la aplicación (*Application Name*):** Nombre de la aplicación tal y como se mostrará a los usuarios. Para este proyecto, se utiliza *MyCinemApp*.

- **Nombre del proyecto (*Project Name*):** Nombre del directorio que se creará para el proyecto y el nombre con el que se le hará referencia en Eclipse.

- **Nombre del paquete (*Package Name*):** Nombre del paquete a utilizar para la aplicación (siguiendo las mismas normas que los paquetes en el lenguaje de programación Java). El nombre de paquete debería ser único, distinto del resto de paquetes instalados en el sistema Android. Para este proyecto, se ha utilizado el nombre del paquete: com.irmg.mycinemapp.

- **SDK mínimo requerido (*Minimum Required SDK*):** Versión mínima de Android que soporta la aplicación que se va a desarrollar, especificada usando los Niveles API. Para que la aplicación funcione en el mayor número de dispositivos posible, se establece este valor a la mínima versión que permita implementar las características principales de la aplicación. Si hay alguna funcionalidad que puede implementarse únicamente en versiones nuevas de Android y no es crítica, se puede habilitar sólo si se está ejecutando una versión que la soporte. En el presente proyecto se ha utilizado la versión API 8: Android 2.2 (Froyo) para que la aplicación funcione aproximadamente en un 93 % de dispositivos.
- **SDK objetivo (*Build SDK*):** Versión más reciente de Android con la que se va a compilar y probar la aplicación (de nuevo usando el Nivel API). Suele tratarse de la versión más reciente o la primera que soporte todas las APIs a las que se desea acceder. En el presente proyecto se ha utilizado la versión Android 2.2 (API 8).

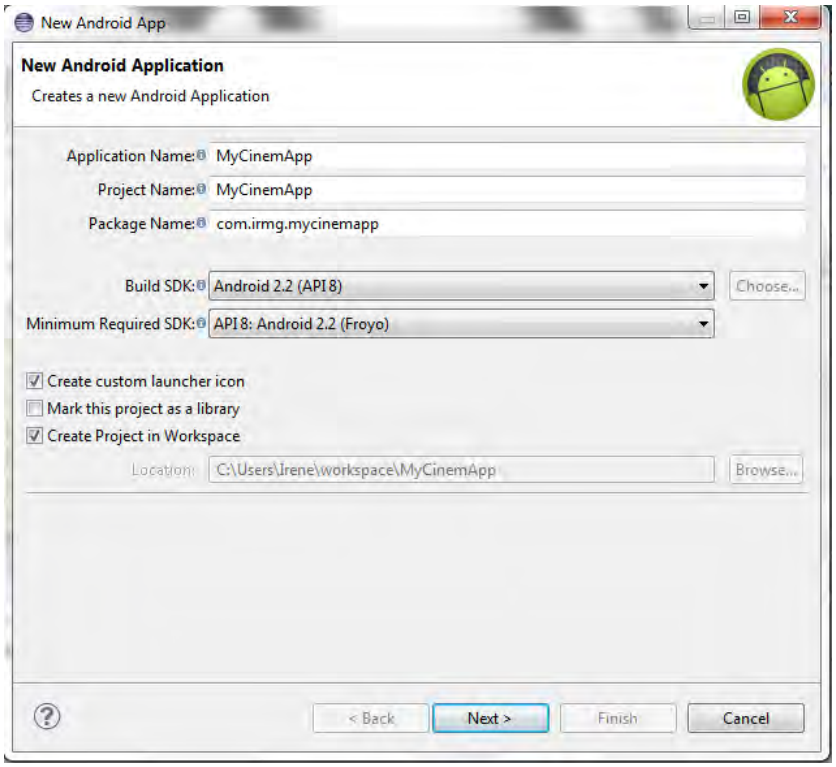


Figura 3.12: Formulario para la creación de un proyecto Android en Eclipse

Una vez rellenados dichos campos, se deja el resto de los campos a los valores por defecto y se pulsa *Next*. La ventana a la que es dirigido el usuario posteriormente ayuda a crear un icono para el lanzador de la aplicación. Se puede personalizar el icono de distintas maneras y la herramienta generará un icono para todas las densidades de pantalla. Antes de publicar la aplicación, el desarrollador debe asegurarse de que su icono sigue las especificaciones definidas en la guía de diseño de Iconografía[1]. A continuación, se pulsa *Next*. En la siguiente ventana, se puede seleccionar una plantilla de actividad con la que empezar a construir la aplicación. Para comenzar, se selecciona *BlankActivity* (actividad vacía). A continuación se pulsa *Next*. Finalmente, aparece un nuevo formulario en el que se mantienen todos los detalles de la actividad en sus valores por defecto y se pulsa *Finish*.

[1] `https://developer.android.com/design/style/iconography.html`

Estructura de un proyecto Android en Eclipse

Al crear un proyecto nuevo con el entorno de desarrollo eclipse, se genera una serie de carpetas. La estructura del proyecto se muestra en la Figura 3.13.

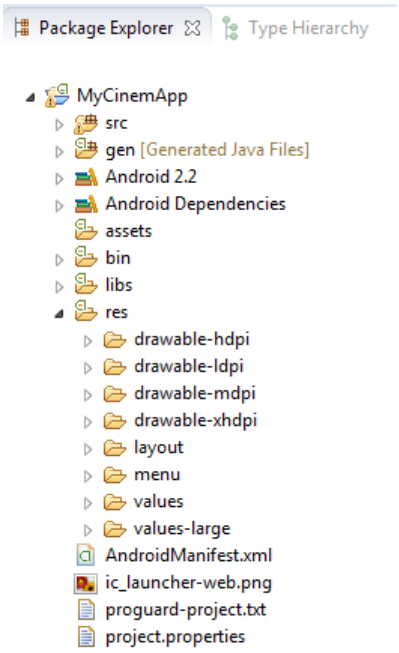


Figura 3.13: Estructura del Proyecto Android en Eclipse

A continuación se describe la utilidad de las carpetas y ficheros generados:

- Carpeta *src*: Almacena todas las clases de la aplicación.
- Carpeta *res*: Guarda todos los recursos que necesita la aplicación (imágenes, XML de interfaz, XML de *strings*, sonidos, audios, etc). Android asigna a cada recurso una dirección y lo define como un atributo de una clase llamada R, contenida dentro de la carpeta gen. La carpeta res contiene las carpetas que se listan a continuación.
  - Carpetas *drawable-hdpi*, *drawable-mdpi*, *drawable-ldpi*: Guardan las imágenes de la aplicación. Las extensiones (*high*, *medium* y *low*) hacen referencia al tipo de densidad de pantalla del dispositivo o emulador. Al abrir cualquiera de las tres carpetas aparece la imagen ic_launcher.png que corresponde al icono de la aplicación.
  - Carpeta *layout*: Contiene todos los ficheros XML que definen la interfaz de las actividades de la aplicación.
  - Carpeta *values*: Contiene el fichero strings.xml en el que se declaran todas las cadenas de texto que se vayan a utilizar en la aplicación.
- Carpeta *gen*: Contiene la clase R.java que referencia todos los recursos de la aplicación. Dentro de la clase R.java, cada subcarpeta de la carpeta *res* está declarada como una clase *static* y cada recurso de cada carpeta como un atributo *static* de su correspondiente clase. De esta forma, se accede a cualquier recurso de la aplicación y desde cualquier clase de la forma R.subcarpeta_res.nombre_recurso.
- Fichero AndroidManifest.xml: Contiene todos los permisos de la aplicación (permisos para utilizar wifi, bluetooth, gps, etc) y todas las actividades que se vayan creando.

Ejecución de la aplicación Android en Eclipse

La Tabla 3.3 resume las alternativas para ejecutar una aplicación Android en Eclipse.

Alternativas	Proceso de Ejecución
Ejecución de la aplicación en un AVD (*Android Virtual Device*)	Para crear un AVD: 1. Ejecutar el Gestor de dispositivos virtuales de Android. En Eclipse, se pulsa el botón de la barra de herramientas *Android Virtual Device Manager*  . 2. En el panel Android Virtual Device Manager, pulsar sobre *New*. 3. Se rellenan los detalles para el AVD: nombre, una plataforma objetivo, un tamaño para la tarjeta SD, y una piel (por defecto es HVGA). 4. Pulsar *Create AVD* (Crear AVD). 5. Selecciona el nuevo AVD en el Gestor de dispositivos virtuales de Android y pulsar en *Start* (Iniciar). 6. Una vez haya terminado de arrancar el emulador, desbloquear la pantalla. Para ejecutar la aplicación desde Eclipse: 1. Se selecciona la opción del menú *Run->Run Configurations-> Android Application*. 2. En la esquina superior izquierda, pulsad sobre el icóno  . En la pestaña Android se selecciona el proyecto Android que se ha creado. En la pestaña *Target*, se selecciona se selecciona el dispositivo virtual que se ha creado. Se pulsa *Apply*. 3. Finalmente, se pulsa *Run*. Eclipse instalará la aplicación en el AVD y más tarde la iniciará.
Ejecución de la aplicación en un dispositivo real	Si se tiene un dispositivo Android, esta es la forma en que se instala y ejecutar la aplicación: 1. Se conecta el dispositivo a la máquina de desarrollo mediante un cable USB. Si se está desarrollando en Windows, es posible que se necesite instalar el driver USB apropiado para el dispositivo. Para obtener ayuda en la instalación de drivers, se puede consultar el documento Drivers USB de los fabricantes[2]. 2. Se habilita la opción Depuración USB en el dispositivo. En la mayoría de los dispositivos que utilizan Android 3.2 o versiones anteriores, se encuentra esta opción en Ajustes-> Aplicaciones-> Desarrollo. En Android 4.0 y superior, esta opción se encuentra en Ajustes->Opciones del desarrollador. Para ejecutar la aplicación desde Eclipse: 1. Se selecciona la opción del menú *Run->Run Configurations-> Android Application*. 2. En la esquina superior izquierda, se pulsa sobre el icóno  . En la pestaña "Android" se selecciona el proyecto Android que se ha creado. En la pestaña *Target*, se seleccionad la opción *Active Devices*. Se pulsa *Apply*. 3. Finalmente, se pulsa *Run*. Eclipse instalará la aplicación en el dispositivo conectado y luego la iniciará.

Tabla 3.3: Proceso de ejecución de la aplicación Android en Eclipse

[2] `http://developer.android.com/tools/extras/oem-usb.html`

Depuración de la aplicación Android en Eclipse

Para depurar una aplicación en Android se coloca un punto de interrupción haciendo doble clic en la barra de la izquierda de la linea de código donde se desea parar la ejecución. Posteriormente se selecciona la opción *Run->Debug* configurations y se define un perfil de ejecución. La aplicación se reiniciará en el emulador, pero esta vez quedará suspendida cuando alcance el punto de interrupción que se ha introducido. Con la tecla F6 se va línea a línea y la tecla F8 continuará hasta el próximo punto de interrupción. Se deberá pulsar el botón  de la esquina superior derecha de eclipse y seleccionar la perspectiva *Debug*.

3.2.2. SQLite. Almacenamiento de los datos requeridos en los módulos Asistente televisivo y Recomendación de películas

SQLite es un motor de bases de datos SQL que ofrece numerosas ventajas, entre ellas: ocupa poco tamaño, no necesita servidor, precisa poca configuración, es transaccional y es de código libre. Android incorpora todas las herramientas necesarias para la creación y gestión de bases de datos SQLite, y entre ellas una API para llevar a cabo de manera sencilla todas las tareas necesarias.

Tal y como se ha descrito en la Sección 3.1.2, se ha utilizado una base de datos SQLite para almacenar la información requerida por los módulos ***Asistente televisivo*** y ***Recomendación de películas***. Dicha base de datos almacena la información consultada por dichos módulos en las siguientes tablas: tabla de gustos televisivos, tablas de programación televisiva, tabla de gustos de cine, y tabla de películas recomendadas al usuario.

3.2.2.1. Panorámica del sistema de gestión de base de datos SQLite

La Tabla 3.4 (“SQLite”, 2014) describe algunas características particulares de la librería SQLite, de código abierto y encargada de implementar un gran subconjunto del estándar SQL-92.

Carácterística	Descripción
Compacta	Con todas las características habilitadas, el tamaño de la librería es inferior a 500KiB. Deshabilitando características opcionales, el tamaño puede quedarse por debajo de los 300KiB. Esto la convierte en una base de datos muy apropiada para usarla en dispositivos con poca memoria, como teléfonos móviles, PDAs y reproductores MP3. Existe una relación entre el uso de memoria y velocidad. Generalmente, SQLite funciona más rápido cuanta más memoria se le reserve. No obstante, el rendimiento de SQLite es suficientemente bueno en entornos de poca memoria.
Autocontenida	La librería SQLite requiere poco soporte de librerías externas o del sistema operativo por lo que es una base de datos adecuada para pequeños dispositivos. Está escrita en ANSI-C y puede compilarse con cualquier compilador de C estándar.
Distribuida bajo dominio público	La libreria SQLite es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin coste.
Fichero único con formato multiplataforma	La base de datos completa (tablas, índices, disparadores y vistas) se almacena en un único fichero, cuyo formato es multiplataforma (Es posible copiar el fichero entre sistemas de 32 y 64 bits o entre arquitecturas *big-endian* y *little-endian*). Esto convierte a SQLite en una buena opción para usarse como un formato de archivos en las aplicaciones.
Manifiesto de tipado	La mayoría de motores SQL utilizan tipado estático. Cada columna de una tabla se asocia con un tipo de datos, y solo pueden introducirse valores de un tipo particular. SQLite elimina esta restricción, y hace que el tipo de datos pueda ser una propiedad del valor en sí, y no de la columna.

Carácterística	Descripción
Registros de longitud variable.	En general, los motores SQL, fijan una cantidad de espacio fija para todas las filas. SQLite, por el contrario, usa sólo la cantidad de disco que necesita para almacenar la información en una fila.
Transaccional	En bases de datos se denomina ACID a un conjunto de características necesarias para que una serie de instrucciones puedan ser consideradas como una transacción. Estás características son: Atomicidad, Consistencia, Aislamiento, y Durabilidad.
Embebido	La mayoría de motores de bases de datos SQL se implementan como un proceso en un servidor separado. Los programas que quieren acceder a la base de datos se comunican con el servidor usando algún tipo de protocolo para enviar peticiones y recibir resultados (normalmente TCP/IP). Por el contrario, con SQLite, el proceso que quiere acceder a la base de datos, lee y escribe directamente en disco, sin necesidad de comunicarse con ningún servidor intermedio. De esta manera se puede hacer una aplicación totalmente autónoma y portable. Esto tiene ventajas y desventajas. La principal ventaja es que no hay ningún servidor que instalar, configurar, inicializar, mantener, etc. Sin embargo, la utilización de un servidor para la base de datos provee mayor protección frente a bugs en el lado de cliente. Además, como el servidor es un único proceso, puede controlar mejor la concurrencia. No obstante, una característica de SQLite es que es la única base de datos sin servidor que permite el acceso de múltiples aplicaciones a la misma base de datos.
No necesita configuración	Debido a que SQLite es un sistema de gestión de base de datos embebido en la aplicación, no necesita instalar ni configurar nada más aparte de la aplicación en cuestión.
Seguridad y Fiabilidad del código	Gran parte del código fuente de SQLite está dedicado a pruebas y verificación. SQLite responde perfectamente a fallos de reserva de memoria, y errores de E/S de disco.

Tabla 3.4: Características de la librería SQLite

A continuación, se resumen las ventajas que se tienen en cuenta a la hora de escoger SQlite como sistema de gestión de base de datos (“SQLite”, 2014), (Cabero y Maldonado, 2007):

- Tamaño: SQLite es una base de datos que ocupa una cantidad muy pequeña de espacio en disco y memoria de manera que es la elección perfecta para crear bases de datos en sistemas operativos para móviles como Android o iOs
- Rendimiento de base de datos: SQLite realiza operaciones de manera eficiente y es más rápido que otros sistemas de gestión de base de datos como MySQL
- Portabilidad: SQLite se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración
- Estabilidad: SQLite es compatible con ACID, reunión de los cuatro criterios de Atomicidad, Consistencia, Aislamiento y Durabilidad
- SQL: SQLite implementa un gran sub-conjunto del estándar SQL-92
- Sin servidor: no tiene un servidor de base de datos ejecutándose en un proceso separado por lo que no es necesario disponer de ningún tipo de conexión
- Simplicidad: SQLite es relativamente simple de aprender a usar
- Coste: SQLite es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin coste y se puede redistribuir libremente

Sin embargo, SQLite es una base de datos con funcionalidad limitada a comparación con otras como MySQL. Su utilización se desaconseja en aplicaciones cliente-servidor, en sistemas en los que se desee utilizar bases de datos de tamaño muy grande y en aplicaciones que requieran alta concurrencia.

3.2.2.2. Utilización de la base de datos SQLite en el desarrollo de aplicaciones para dispositivos móviles Android

A continuación, se explica los conceptos necesarios a la hora de trabajar con una base de datos SQLite en el desarrollo de aplicaciones para para dispositivos móviles Android (Vogel, 2010).

Generalidades a la hora de trabajar con una base de datos SQLite en el desarrollo de aplicaciones para dispositivos móviles Android

El motor de base de datos SQLite se encuentra embebido en los dispositivos Android, por lo que no es necesaria su descarga, configuración y administración. Únicamente es responsabilidad del programador el crear y actualizar la base de datos SQLite, ya que de su administración se encarga la plataforma Android. Tampoco es necesario incluir librerías adicionales para trabajar con una base de datos SQLite ya que todas las librerías necesarias pertenecen al kit de desarrollo de aplicaciones en Android.

SQLite, utiliza el lenguaje SQL para las consultas (SELECT), manipulación de datos (INSERT, DELETE, etc.), y de definición de datos (CREATE TABLE, etc).

SQLite presenta unas pequeñas variaciones donde se desvía del estándar SQL-92, que aplica para la mayoría de bases de datos SQL. Entre ellas, el uso de FOREIGN KEY, transacciones anidadas, RIGHT OTHER JOIN, FULL OTHER JOIN y algunos usos de ALTER TABLE no son válidos en SQLite.

SQLite soporta los tipos de datos TEXT (similar a los String en Java), INTEGER (similar a Integer en Java) y REAL (similar a Double en Java). Si se utiliza cualquier otro tipo de dato, se convierten de manera automática a estos tres tipos de datos.

Las bases de datos en SQLite tienen un nivel de acceso privado que abarca únicamente la aplicación en la que fueron creadas. Al crear una base de datos durante el desarrollo de la aplicación, esta se guarda en el directorio DATA/data/APP_NAME/databases/FILENAME, donde:

- DATA: Ruta que retorna el método Environment.getDataDirectory(). Dicho método regularmente apunta a la tarjeta SD como ubicación predefinida.
- APP_NAME: Nombre de la aplicación
- FILENAME: Nombre que se le da a la base de datos al crearla

Paquetes necesarios para trabajar con una base de datos SQLite en Android

El paquete android.database ("API", 2014) contiene todas las clases necesarias para trabajar con bases de datos. El paquete android.database.sqlite ("API", 2014) contiene las clases especificas de SQLite.

Crear y actualizar la base de datos SQLite mediante la utilización de la clase SQLiteOpenHelper

Para crear y actualizar una base de datos en Android, se crea una clase propia que herede de la clase SQLiteOpenHelper ("API", 2014) e implemente las necesidades concretas de la aplicación desarrollada.

El constructor de la clase SQLiteOpenHelper invoca al método super(), al que se le especifica el nombre y versión de la base de datos. Además, dentro de dicha clase, se sobrescriben los métodos onCreate() para crear la base de datos, y onUpgrade() para actualizar la base de datos en caso de que existan cambios en el esquema de la misma. Ambos métodos reciben como parámetro un objeto SQLiteDatabase, siendo este objeto la representación de una base de datos en Java. Dichos métodos vienen descritos en la Tabla 3.5.

Método	Descripción
onCreate(SQLiteDatabase db)	Se ejecuta automáticamente cuando es necesaria la creación de la base de datos, es decir, cuando aún no existe. Las tareas que se realizan en este método son la creación de todas las tablas necesarias y la inserción de los datos iniciales si son necesarios. Para la creación de las tablas se utiliza la sentencia SQL: ▪ CREATE TABLE nombre_tabla(columna1 tipo PRIMARY KEY(una o más columnas), columna2 tipo, columna3 tipo, ... columnaN tipo,);
onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)	Se lanza automáticamente cuando es necesaria una actualización de la estructura de la base de datos o una conversión de los datos. Para ello, como parámetros recibe la versión actual de la base de datos en el sistema, y la nueva versión a la que se quiere convertir. En la aplicación desarrollada se ha optado por la opción más sencilla: borrar la tabla actual y volver a crearla con la nueva estructura.

Tabla 3.5: Métodos de la clase *SQLiteOpenHelper* necesarios para crear y actualizar la base de datos SQLite

Para abrir la base de datos desde la aplicación desarrollada, lo primero es crear un objeto de la clase que hereda de SQLiteOpenHelper que se ha creado, pasándole como parámetro el contexto de la aplicación (una referencia a la actividad principal), el nombre de la base de datos, un objeto CursorFactory que típicamente no será necesario (al no ser necesario se le asigna el valor *null*), y por último la versión de la base de datos. La simple creación de este objeto puede tener varios efectos:

- Si la base de datos ya existe y su versión actual coincide con la solicitada simplemente se realiza la conexión con ella.
- Si la base de datos existe pero su versión actual es anterior a la solicitada, se llama automáticamente al método onUpgrade() para convertir la base de datos a la nueva versión y se conectará con la base de datos convertida.
- Si la base de datos no existe, se llamará automáticamente al método onCreate() para crearla y se conectará con la base de datos creada.

Una vez se tiene la referencia al objeto de la clase que hereda de SQLiteOpenHelper, se llama a su método getReadableDatabase() o getWritableDatabase() para obtener una referencia a la base de datos en modo lectura o escritura, respectivamente.

Manipular la base de datos SQLite mediante la utilización de la clase SQLiteDatabase

La clase SQLiteDatabase ("API", 2014) provee los métodos necesarios para manipular una base de datos SQLite. La Tabla 3.6 describe algunos de los métodos pertenecientes a dicha clase que permiten la manipulación de los datos de la base de datos SQLite y que se utilizan en la aplicación desarrollada.

Método	Descripción
long insert(String table, String nullColumnHack, ContentValues values)	Permite insertar una fila en una tabla de la base de datos SQLite. El parámetro *table* especifica el nombre de la tabla. El objeto ContentValues permite definir una clave correspondiente a una columna determinada y un valor para dicha columna. El párametro *nullColumnHack* es opcional y permite especificar el nombre de la columna en

Método	Descripción
	la que se puede asignar un valor nulo si el objeto ContentValues está vacío. De esta manera, se insertará una fila a pesar de no indicarse ningún valor para ninguna columna, insertando un valor nulo en la columna especificada. El parámetro *nullColumnHack* puede valer *null* y en dicho caso, no se inserta una fila si el objeto ContentValues está vacío. El método devuelve el identificador de la fila insertada o -1 si ha ocurrido algún error.
int update(String table, ContentValues values, String whereClause, String[] whereArgs)	Permite actualizar filas de una tabla de la base de datos SQLite. El parámetro table específica el nombre de la tabla. El objeto ContentValues permite definir una clave correspondiente a una columna determinada y el valor con el que actualizar dicha columna. El parámetro *whereClause* permite especificar opcionalmente la clausula WHERE que permite seleccionar que filas se actualizarán (si se asigna un valor *null* se actualizarán todas las filas de la tabla). Se puede incluir la cadena "?s" en la clausula WHERE que se reemplazará con los valores especificados por el parámetro *whereArgs*. El método devuelve el número de filas afectadas.
int delete(String table, String whereClause, String[] whereArgs)	Permite borrar de una tabla de la base de datos SQLite. El parámetro table específica el nombre de la tabla. El parámetro *whereClause* permite especificar opcionalmente la clausula WHERE que permite seleccionar que filas se borraran (si se asigna un valor null se borran todas las filas de la tabla). Se puede incluir la cadena "?s" en la clausula WHERE que se reemplazará con los valores especificados por el parámetro *whereArgs*. El método devuelve el número de filas afectadas.
void execSQL(String sql)	Permite ejecutar cualquier sentencia SQL(exceptuando SELECT o cualquier sentencia que devuelva datos) pasadas como parámetro directamente.

Tabla 3.6: Métodos de la clase SQLiteDatabase utilizados en la aplicación desarrollada

En cuanto a las consultas a la base de datos, se han implementado mediante el método rawQuery (String sql, String[] selectionArgs), también de la clase SQLiteDatabase, que permite ejecutar una consulta SQL (SELECT) en la base de datos. El primer parámetro especifica la consulta SQL a realizar. Se puede incluir la cadena "?s" en la clausula WHERE que se reemplazará con los valores especificados por el parámetro *selectionArgs*.

La consulta devuelve un objeto de la clase Cursor ("API", 2014) que apunta a una posición anterior a la primera entrada de la tabla que satisface la consulta. Para conocer el número de elementos que ha retornado la consulta se utiliza el método getCount(). Para moverse entre las filas individuales de datos se utiliza los métodos moveToFirst() y moveToNext(). A través del método isAfterLast() se puede comprobar si aún existen datos. El método getString(int columnIndex) devuelve el valor en formato String de la columna especificada por el parámetro *columnIndex*. Dichos métodos pertenecen a la clase Cursor.

3.2.3. MySQL y PHP. Almacenamiento de datos en el módulo Registro e identificación de usuario

En la presente Sección se realiza una descripción de las tecnologías utilizadas en el desarrollo del módulo ***Registro e identificación de usuario***, cuya función principal es la de almacenar en una base de datos MySQL remota una tabla de usuarios registrados. Esta tabla almacena para cada usuario el nombre, la contraseña, las preferencias televisivas y los gustos de cine. La razón por la cual se ha decidido almacenar dicha información en una tabla de una base de datos MySQL remota, es que mediante este sistema el usuario puede iniciar sesión desde cualquier dispositivo mediante su nombre y contraseña y acceder a su información de usuario.

Por el momento, el API de Android no provee ningún mecanismo que permita acceder directamente a través de Internet a una base de datos MySQL remota y ejecutar una consulta dentro de ella. Para poder acceder a la base de datos remota MySQL, se ha desarrollado un servicio web.

La aplicación Android (cliente) envía una consulta mediante HTTP a un servidor web que accede a la base de datos MySQL y devuelve la información solicitada a la aplicación Android en formato JSON. Por lo tanto, el módulo ***Registro e identificación de usuario*** está dividido en dos partes: La parte web y la parte Android.

- Parte web: Para la implementación de la parte web se utiliza como lenguaje en la parte de servidor PHP, como base de datos MySQL, y como servidor web Apache. Para ello, se ha utilizado ***x10hosting*** ("x10hosting", 2014), un servidor web que incluye bases de datos MySQL, un servidor web Apache y el intérprete para lenguaje de script PHP 5. Además, proporciona acceso al administrador de base de datos: ***PhpMyAdmin***.

- Parte Android: la aplicación Android (cliente) se conecta y envía una consulta al servidor mediante HTTP y obtiene una respuesta en formato JSON.

3.2.3.1. Servidor web: x10hosting

Como servidor web para el almacenamiento de los ficheros PHP y de las bases de datos de la aplicación se ha utilizado ***x10hosting***. A continuación, se listan las características más importantes que tiene este servidor para su plan gratuito (sin publicidad) ("Plan comparison", 2014):

- cPanel 11+.
- 1GB de almacenamiento.
- 10 Gb de banda ancha.
- Cuentas FTP.
- Cuentas de e-mail.
- Posibilidad de agregar dominios o subdominios.
- Servidor Apache 2.2.
- PHP 5.4 y 2 bases de datos MySQL 5.
- SMTP, CGI , Perl, Python, PHP Sendmail
- Auto-instalador de scripts.
- Soporte técnico a través de foros.
- Acceso al administrador de base de datos *phpMyAdmin 4.1.8* desde el panel de control

Utilización de x10hosting en el módulo *Registro e identificación de usuario*

Para desarrollar el módulo ***Registro e identificación de usuario*** se utilizan las siguientes utilidades proporcionadas por el servidor ***x10hosting***: el panel de control cPanel 11+, el servidor web Apache 2.2, las bases de datos MySQL, el intérprete para PHP 5.4 y el administrador de base de datos *phpMyAdmin* 4.1.8.

La Figuras 3.14-3.16 muestran capturas de pantalla de las características utilizadas del panel de control de *x10hosting*. En primer lugar, la Figura 3.14 muestra el administrador de ficheros del panel de control, que se encarga de almacenar todos los ficheros PHP necesarios en el módulo ***Registro e identificación de usuario.*** A continuación, la Figura 3.15 muestra el nombre del dominio que se ha creado y asociado a la cuenta del servidor *x10hosting*: proyectoquetver.x10.mx.

Finalmente, la Figura 3.16 muestra las siguientes bases de datos:

- **proyec36_questionario**: base de datos para almacenar los datos del cuestionario de evaluación (Capítulo 5)

- **proyec36_webservice**: base de datos que contiene la tabla de usuarios registrados, que se encarga de almacenar el nombre de usuario, la contraseña, y los gustos televisivos y de cine de cada usuario

Tal y como se observa en la Figura 3.16, el panel de control proporciona acceso al administrador de bases de datos *PHPMyAdmin*.

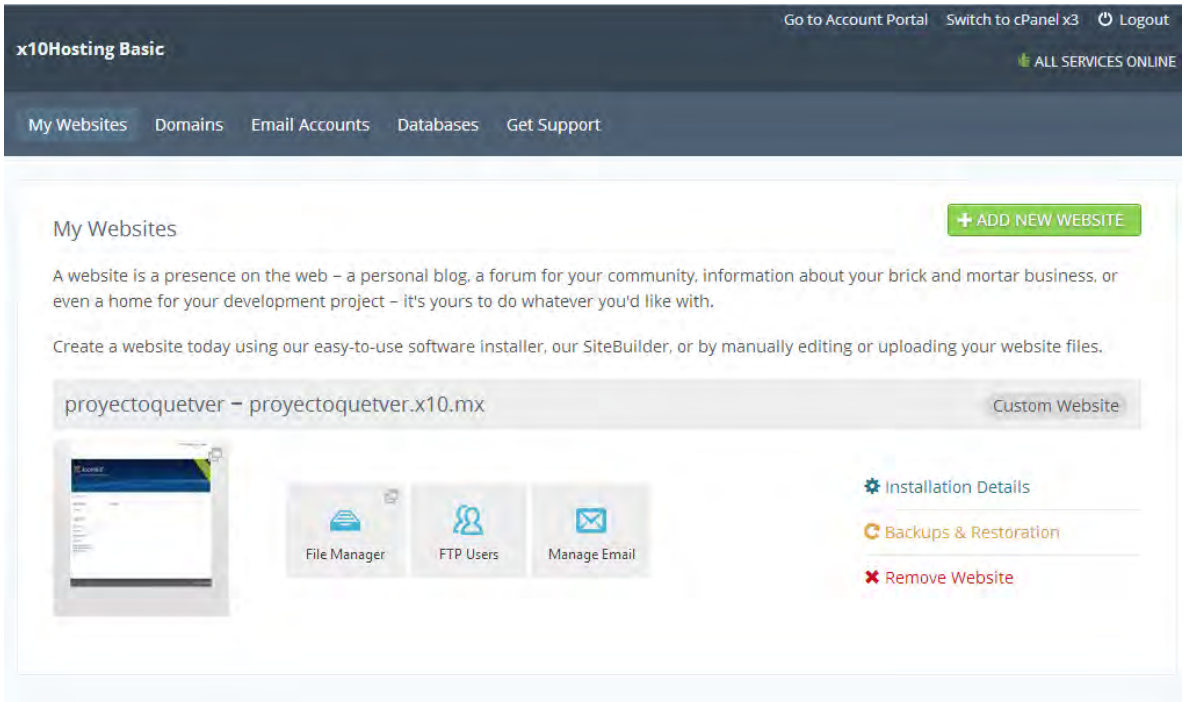


Figura 3.14: Panel de Control de x10Hosting: Administrador de ficheros

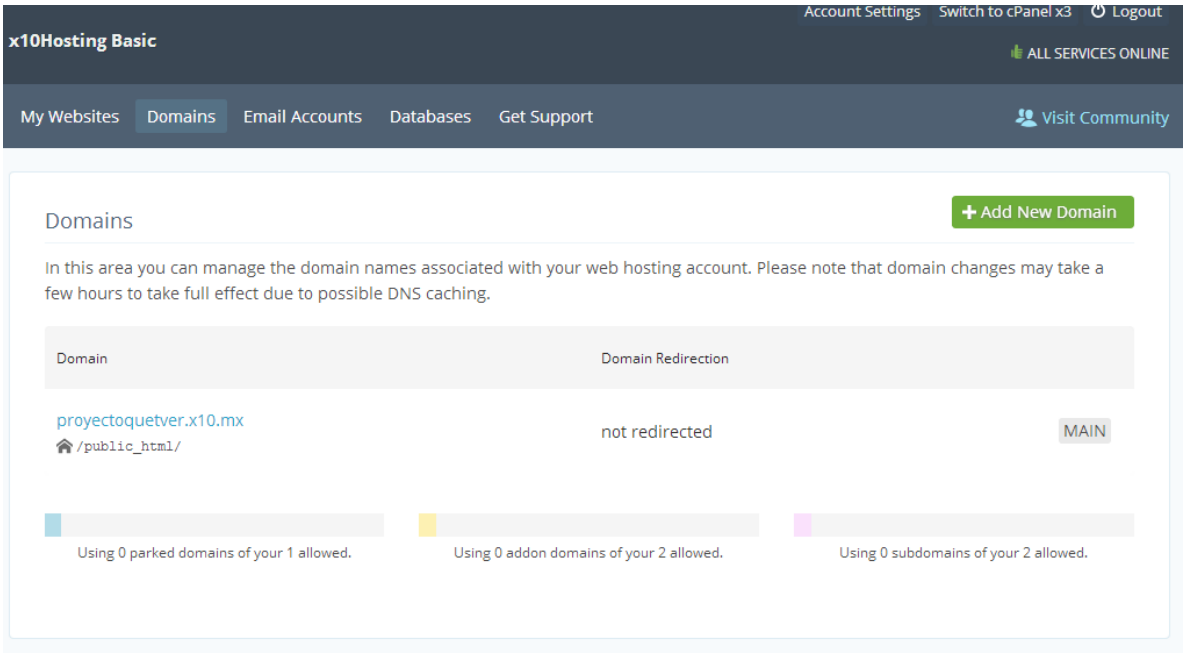


Figura 3.15: Panel de Control de x10Hosting: Dominios

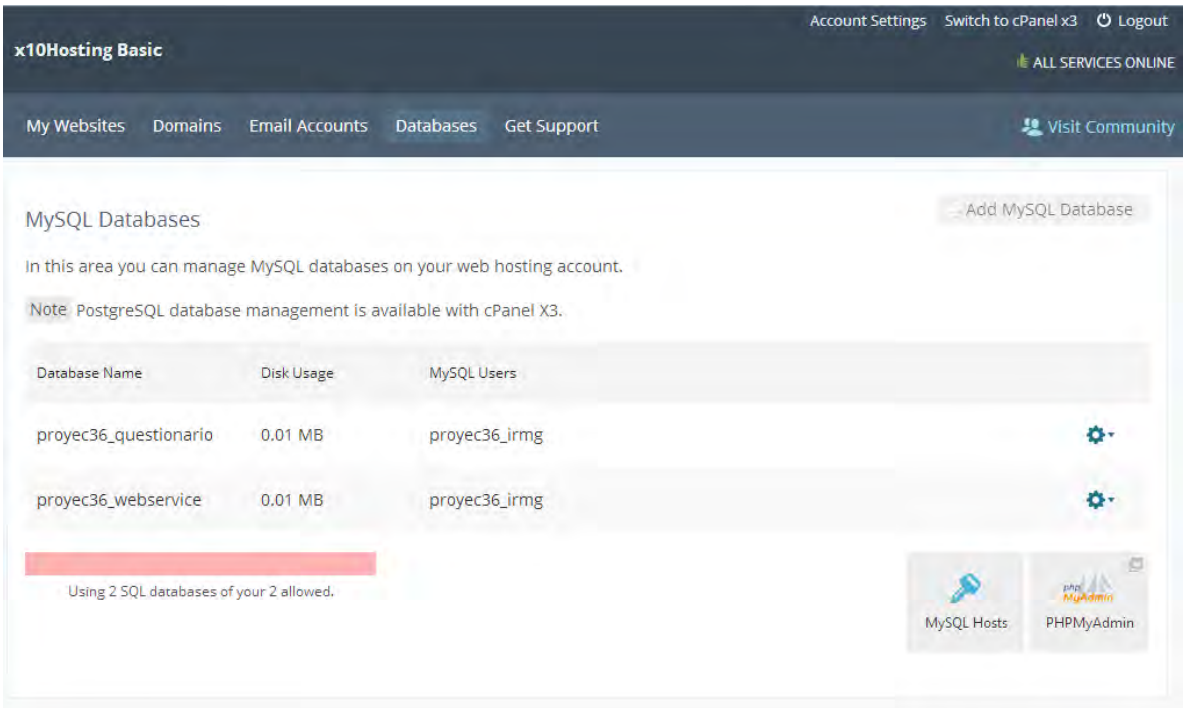


Figura 3.16: Panel de Control de x10Hosting: Bases de datos

3.2.3.2. MySQL

MySQL es un sistema de gestión de bases de datos relacionales desarrollado y distribuido por la compañía comercial MySQL AB. El software de base de datos MySQL proporciona un servidor de base de datos SQL (*Structured Query Language*) muy rápido, multihilo, multiusuario y robusto. El servidor MySQL está diseñado para entornos de producción críticos, con alta carga de trabajo así como para integrarse en software para ser distribuido (Oracle, 2013a).

El software MySQL tiene una doble licencia. Los usuarios pueden elegir entre usar el software MySQL como un producto *Open Source* bajo los términos de la licencia GNU *General Public License* ("GPL", 2014) o pueden adquirir una licencia comercial estándar de MySQL AB. La Tabla 3.7 (Oracle, 2013a) resume las características más importantes del software de base de datos MySQL.

Característica	Descripción
Interioridades y portabilidad	■ Escrito en C y en C++ ■ Probado con un amplio rango de compiladores diferentes ■ Funciona en diferentes plataformas ■ APIs disponibles para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, y Tcl ■ Utilización de multihilos mediante hilos del kernel ■ Proporciona sistemas de almacenamiento transaccionales y no transaccionales ■ Usa tablas en disco *B-tree* muy rápidas con compresión de índice ■ Relativamente sencillo de añadir otro sistema de almacenamiento ■ Un sistema de reserva de memoria muy rápido basado en hilos ■ *Joins* muy rápidos usando un *multi-join* de un paso optimizado ■ Tablas *hash* en memoria, que son usadas como tablas temporales ■ Las funciones SQL están implementadas usando una librería altamente optimizada y deben ser tan rápidas como sea posible. Normalmente no hay reserva de memoria tras toda la inicialización para consultas. ■ El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor. También está disponible como biblioteca y puede ser incrustado en aplicaciones autónomas, que pueden usarse por sí mismas o en entornos donde no hay red disponible. ■ Registros de longitud fija y longitud variable

Característica	Descripción
Tipos de Columnas	▪ Diversos tipos de columnas: enteros con/sin signo de 1, 2, 3, 4, y 8 bytes de longitud, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM, y tipos espaciales *OpenGIS*.
Sentencias y funciones	▪ Soporte completo para operadores y funciones en las cláusulas de consultas SELECT y WHERE ▪ Soporte completo para las cláusulas SQL GROUP BY y ORDER BY. Soporte de funciones de agrupación (COUNT(), COUNT(DISTINCT ...), AVG(), STD(), SUM(), MAX(), MIN(), y GROUP_CONCAT()). ▪ Soporte para LEFT OUTER JOIN y RIGHT OUTER JOIN cumpliendo estándares de sintaxis SQL ▪ Soporte para alias en tablas y columnas como lo requiere el estándar SQL ▪ DELETE, INSERT, REPLACE, y UPDATE devuelven el número de filas que han cambiado (han sido afectadas). Es posible devolver el número de filas que serían afectadas usando un *flag* al conectar con el servidor. ▪ El comando específico de MySQL SHOW puede usarse para obtener información acerca de la base de datos, el motor de base de datos, tablas e índices. El comando EXPLAIN puede usarse para determinar cómo el optimizador resuelve una consulta. ▪ Los nombres de funciones no colisionan con los nombres de tabla o columna. ▪ Puede mezclar tablas de distintas bases de datos en la misma consulta
Conectividad	▪ Los clientes pueden conectar con el servidor MySQL usando sockets TCP/IP en cualquier plataforma ▪ En MySQL 5.0, los servidores Windows soportan conexiones con memoria compartida ▪ La interfaz para el conector J MySQL proporciona soporte para clientes Java que usen conexiones JDBC. Estos clientes pueden ejecutarse en Windows o Unix.

Característica	Descripción
	▪ La interfaz para el conector ODBC (MyODBC) proporciona a MySQL soporte para programas clientes que usen conexiones ODBC (*Open Database Connectivity*). Los clientes pueden ejecutarse en Windows o Unix.
Escalabilidad y límites	▪ Soporte a grandes bases de datos. Se puede utilizar MySQL Server con bases de datos que contienen 60.000 tablas y cerca de 5.000.000.000.000 de registros. ▪ Se permiten hasta 64 índices por tabla. Cada índice puede consistir desde 1 hasta 16 columnas o partes de columnas. El máximo ancho de límite son 1000 bytes.
Seguridad	Un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite verificación basada en el host. Las contraseñas son seguras porque todo el tráfico de contraseñas está cifrado cuando se conecta con un servidor.
Localización	▪ El servidor puede proporcionar mensajes de error a los clientes en muchos idiomas ▪ Soporte completo para distintos conjuntos de caracteres ▪ Todos los datos se guardan en el conjunto de caracteres elegido ▪ Todas las comparaciones para columnas normales de cadenas de caracteres son *case-insensitive* ▪ La ordenación se realiza acorde al conjunto de caracteres elegido (usando colación Sueca por defecto). Es posible cambiarla cuando arranca el servidor MySQL.
Clientes y herramientas	▪ MySQL server tiene soporte para comandos SQL para chequear, optimizar, y reparar tablas ▪ Todos los programas MySQL pueden invocarse con las opciones *–help* o -? para obtener asistencia en línea

Tabla 3.7: Carácteristicas de MySQL

Utilización de MySQL en el módulo *Registro e identificación de usuario*

Tal y como se ha explicado, *x10hosting* incluye el sistema de gestión de base de datos MySQL 5. En el desarrollo del módulo ***Registro e identificación de usuario***, se ha creado la base de datos MySQL ***proyec36_webservice*** cuyo objetivo principal es el de almacenar la información de usuario en la tabla ***usuarios***.

A continuación se listan y describen las columnas que forman dicha tabla:

- ***id***: Identificador de usuario. Contiene un número entero que se autoincrementa a medida que se añade una fila. Es de tipo INT. Tiene la restricción UNIQUE que sirve para indicar que una columna tendrá valores únicos para cada fila de la tabla.
- ***usuario***: Nombre de usuario. Es de tipo VARCHAR. Esta columna está definida como clave primaria (PRIMARY KEY), que indica que la columna tendrá valores únicos para cada fila de la tabla y que no podrá tener valores nulos.
- ***password***: Contraseña del usuario. Es de tipo VARCHAR.
- ***deportes***, ***cine***, ***documentales***, ***musica***, ***noticias***, ***series***, ***programas***, ***realities***, ***concursos***: Almacenan un valor de tipo TEXT con los gustos televisivos del usuario. '1' si el usuario desea obtener recomendaciones de la opción correspondiente, '0' en caso contrario.
- ***genero***: almacena el género de cine sobre el usuario desea obtener recomendaciones de películas. Es de tipo TEXT.
- ***pais***: almacena el pais origen de las películas de cine sobre el usuario desea obtener recomendaciones de películas. Es de tipo TEXT.
- ***desde***: almacena la fecha desde la cual se desea obtener resultados en las recomendaciones de cine.
- ***hasta***: almacena la fecha hasta la cual se desea obtener resultados en las recomendaciones de cine.
- ***excluir_series***: almacena un valor de tipo TEXT que representa si el usuario desea excluir las series de las recomendaciones de películas. '1' si el usuario desea obtener recomendaciones de series, '0' en caso contrario.
- ***excluir_documentales***: almacena un valor de tipo TEXT que representa si el usuario desea excluir los series de las recomendaciones de películas. '1' si el usuario desea obtener recomendaciones de series, '0' en caso contrario.

Para algunas columnas (i.e. *deportes*, *cine*, *documentales*, *musica*, *noticias*, *series*, *programas*, *realities*, *concursos*, *excluir_series* y *excluir_documentales*), se ha utilizado el tipo TEXT y no el tipo BIT con el objetivo de mantener coherencia con la base de datos SQLite ya que esta únicamente soporta los tipos de datos TEXT, INTEGER y REAL.

La creación de la base de datos *proyec36_webservice* y de la tabla *usuarios* se ha realizado desde la interfaz del administrador de base de datos *PhpMyAdmin*. La manipulación de los datos se ha realizado desde los ficheros PHP almacenados en el servidor, y para ello, se ha utilizado sentencias de manipulación de datos (*Data Manipulation Statements*) SQL (Oracle, 2013b) como SELECT, INSERT, UPDATE, DELETE, etc.

3.2.3.3. PhpMyAdmin

PhpMyAdmin (`http://www.phpmyadmin.net`) es un administrador de bases de datos MySQL que se utiliza vía web. *PhpMyAdmin* permite realizar casi todas las operaciones MySQL: crear o eliminar bases de datos; crear, eliminar o alterar tablas; eliminar, editar o agregar campos; ejecutar consultas SQL, etc. *PhpMyAdmin* permite realizar las operaciones MySQL directamente a través de su interfaz o mediante la ejecución de la sentencia SQL.

Las principales características de phpMyAdmin son:

- Interfaz gráfica muy intuitiva
- Soporta gran parte de las características de MySQL
- Multiplataforma
- Multilenguaje (72 lenguajes)
- Se encuentra disponible bajo la licencia GPL
- Escrito en PHP

Utilización de PhpMyAdmin en el módulo *Registro e identificación de usuario*

El servidor *x10hosting* proporciona acceso al administrador de bases de datos *phpMyAdmin 4.1.8*. Se ha utilizado dicha herramienta para administrar y gestionar la base de datos *proyec36_webservice*, encargada de almacenar la tabla *usuarios*.

La Figura 3.17 muestra la intuitiva interfaz gráfica de *phpMyAdmin*. En ella se observa la tabla *usuarios* sobre la que se puede realizar operaciones SQL a través de la interfaz.

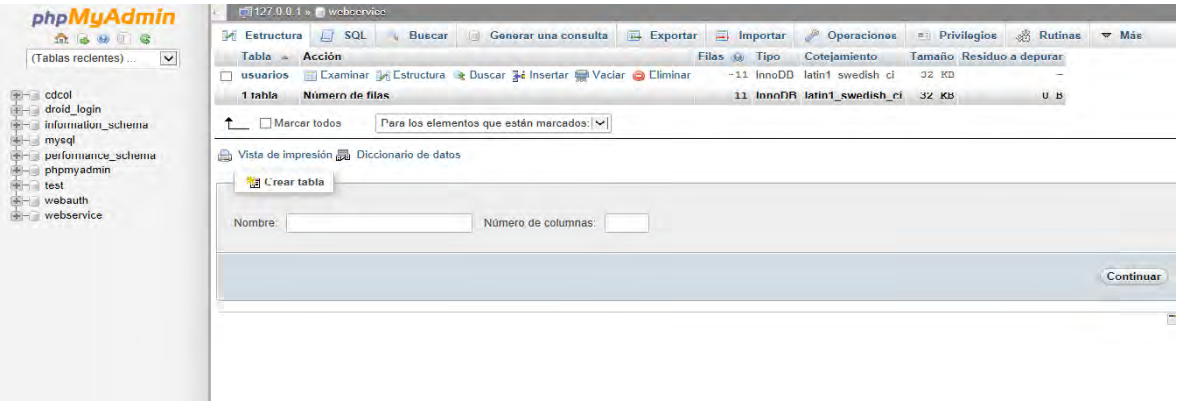


Figura 3.17: Interfaz gráfica *PhpMyAdmin*

3.2.3.4. PHP

PHP (acrónimo recursivo de PHP: *Hypertext Preprocessor*) (Achour et al., 2013) es un lenguaje de código abierto utilizado en el desarrollo web y que puede ser incrustado en HTML. Principalmente, PHP está enfocado a la programación de scripts en el lado del servidor.

PHP puede usarse en la mayor parte de sistemas operativos, incluyendo Linux, muchas variantes de Unix (incluyendo HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X y RISC OS. Además, PHP admite la mayoría de servidores web, incluyendo Apache, y ofrece la posibilidad de utilizar programación por procedimientos, programación orientada a objetos (POO), o una mezcla de ambas.

Una de las características más potentes y destacables de PHP es su soporte para un amplio abanico de bases de datos. Para acceder a una base de datos MySQL desde la página web desarrollada, se puede utilizar cualquiera de las dos opciones descritas a continuación:

- Utilizar las extensiones de bases de datos específicas del proveedor: PHP ofrece varios controladores y complementos de MySQL para acceder y manejar MySQL.
- La extensión Objetos de Datos de PHP (PDO-*PHP Data Objects*): define una interfaz ligera para poder acceder a bases de datos en PHP. No se puede realizar ninguna de las funciones de la base de datos utilizando la extensión PDO por sí misma sino que se debe utilizar un controlador de PDO específico de la base de datos para tener acceso a un servidor de bases de datos. PDO proporciona una capa de abstracción de acceso a datos, lo que significa que, independientemente de la base de

datos que se esté utilizando, se usan las mismas funciones para realizar consultas y obtener datos. Sin embargo, PDO no proporciona una abstracción de bases de datos ya que no reescribe SQL ni emula características ausentes. PDO_MYSQL es un controlador que implementa la interfaz de Objetos de Datos de PHP (PDO) para permitir el acceso de PHP a bases de datos de MySQL 3.x, 4.x y 5.x. Dicho controlador, aprovecha el soporte nativo de sentencias preparadas presente en MySQL 4.1 y superior.

Utilización de PHP en el módulo *Registro e identificación de usuario*

El servidor *x10hosting* incluye el intérprete para el lenguaje de programación PHP 5.4. Para el acceso a la base de datos MySQL se ha decidido utilizar el controlador que implementa la interfaz de Objetos de Datos de PHP (PDO) para permitir el acceso de PHP a la base de datos de MySQL 5.x.

Se han desarrollado los siguientes ficheros PHP en el lado del servidor:

- **config.inc.php**: realiza la configuración necesaria para acceder a la base de datos MySQL y posteriormente, realiza la conexión a dicha base de datos.
- **registro.php**: realiza el registro del usuario en la tabla *usuarios* de la base de datos *proyec36_webservice*. El usuario debe introducir su nombre de usuario, contraseña y sus gustos televisivos y de cine. Devuelve si la operación se ha llevado a cabo con éxito. Incluye el fichero *config.inc.php* para poder conectarse a la base de datos MySQL.
- **identificacion.php**: realiza la identificación del usuario. Comprueba que el usuario esta registrado en la base de datos y devuelve si la operación se ha llevado a cabo con éxito. Incluye el fichero *config.inc.php* para poder conectarse a la base de datos MySQL.
- **gustos.php**: devuelve en formato JSON los gustos televisivos y de cine del usuario especificado. Incluye el fichero *config.inc.php* para poder conectarse a la base de datos MySQL.
- **modificar_gustos.php**: permite al usuario modificar los gustos televisivos que ha registrado. Incluye el fichero *config.inc.php* para poder conectarse a la base de datos MySQL.

3.2.3.5. HTTP

HTTP (*Hypertext Transfer Protocol*), desarrollado por W3C e IETF, es un protocolo de nivel de aplicación de la web que sigue el esquema petición-respuesta entre un cliente y un servidor. HTTP lleva utilizándose desde 1990 y ha pasado por múltiples versiones del protocolo, siendo HTTP/1.1 (Fielding et al., 1999) el estándar actual.

La información accedida mediante HTTP se la llama recurso y se identifica mediante un localizador uniforme de recursos (URL). Los recursos pueden ser archivos, el resultado de la ejecución de un programa, una consulta a una base de datos, etc.

Una solicitud HTTP está formada por los siguientes campos:

- Método que se aplica al recurso
- Identificador del recurso (URL)
- Protocolo y versión que se utiliza para realizar la petición
- Cabecera: el uso de campos en la cabecera enviados en las transacciones HTTP le dan flexibilidad al protocolo. Estos campos permiten que se envíe información descriptiva en la transacción, permitiendo así la autenticación, cifrado e identificación de usuario.
- Contenido, si procede.

Existen 8 métodos para las solicitudes en HTTP 1.1. Los más importantes son:

- GET: Solicita el recurso al servidor

- POST: Envío de datos al servidor (Ej. formularios)
- HEAD: Lectura de cabeceras (mismo que GET pero no devuelve el contenido del recurso)

Por otro lado, la respuesta HTTP está formada por los siguientes campos:

- Protocolo y versión
- Código de respuesta que indica si la petición fue correcta o no
- Cabecera: permite que se envíe información descriptiva en la transacción
- Contenido de recurso, si procede

A continuación se listan los códigos de respuesta:

- 2XX: Éxito (Ej. 200 OK)
- 3XX: Redirecciones (Ej. 301 *Moved permanently*)
- 4XX: Errores provocados por el cliente (Ej. 404 *Not Found*)
- 5XX: Errores provocados por el servidor (Ej. 500 *Internal Server Error*)

Utilización de HTTP en el módulo *Registro e identificación de usuario*

Para enviar peticiones HTTP al servidor Apache con consultas a la base de datos MySQL, se ha utilizado en la aplicación Android (cliente) las clases descritas en la Tabla 3.8 ("HttpComponents", 2014). Mediante la utilización de dichas clases se puede crear un consumidor de un servicio Web.

Clase	Paquete	Descripción	Métodos utilizados
DefaultHttpClient	org.apache.http.impl.client	Cliente utilizado para enviar las peticiones al servidor	HttpResponse execute(HttpUriRequest request) : ejecuta la petición al servidor y devuelve la respuesta del servidor.
UrlEncodedFormEntity	org.apache.http.client.entity.	Permite enviar una lista de pares nombre/valor al servidor contenidos en una entidad. Muy útil para utilizar en métodos POST.	UrlEncodedFormEntity(List<? extends NameValuePair> parameters, String encoding): Constructor de la entidad que contiene una lista de pares nombre/valor a enviar al servidor en el método POST. La instancia de dicha clase se pasa como parámetro al método setEntity() de la clase HttpPost.
HttpGet	org.apache.http.client.methods	Método GET del protocolo HTTP	HttpPost(String uri): Constructor a partir de la URL. La instancia creada se pasa como parámetro al método execute() invocado por la instancia de la clase DefaultHttpClient.
HttpResponse	org.apache.http	Procesa la respuesta enviada por el servidor.	HttpEntity getEntity(): Devuelve el objeto de la clase HttpEntity que representa los datos contenidos en la respuesta del servidor.

Clase	Paquete	Descripción	Métodos utilizados
HttpPost	org.apache.http.client.methods	Método POST del protocolo HTTP	▪ HttpPost(String uri): Constructor a partir de la URL. La instancia creada se pasa como parámetro al método execute() invocado por la instancia de la clase DefaultHttpClient. ▪ void setEntity(HttpEntity entity): asocia al método POST con la entidad que contiene datos a enviar al servidor. En el caso de la aplicación desarrollada, se ha pasado como parámetro una instancia de la clase UrlEncodedFormEntity.
HttpEntity	org.apache.http	Representa los datos enviados con una petición o una respuesta	InputStream getContent(): Devuelve un objeto *InputStream* con el contenido de los datos.

Tabla 3.8: Clases del Paquete org.apache.http

Aunque hasta la versión 2.2 de Android (API 8) se puede programar las conexiones de red directamente en el hilo de la UI principal sin que se genere una excepción en el SDK, se va a utilizar la clase AsyncTask que permite ejecutar operaciones en segundo plano e ir publicando los resultados en un hilo de la interfaz gráfica. De esta forma, utilizando programación concurrente es posible realizar tareas de forma concurrente sin realizar operaciones bloqueantes.

La forma básica de utilizar la clase AsyncTask se explica en la Sección 3.3.1. Para conectarse mediante HTTP al servidor web remoto, se crea en primer lugar, una barra de progreso dentro del método onPreExecute(). Todas las operaciones correspondientes a la conexión mediante HTTP con el servidor web y la recepción de la respuesta en formato JSON se realizan dentro del método doInBackground(). Una vez que se obtiene la respuesta en formato JSON del servidor web, dentro del método onPostExecute(), se deshabilita la barra de progreso y se muestra un mensaje por pantalla que indica si la operación se ha llevado a cado con éxito.

Es necesario añadir el permiso de uso para acceso a INTERNET (android.permission.INTERNET) en el fichero AndroidManifest.xml. También es posible utilizar el permiso de uso para acceso al estado de red (android.permission.ACCESS_NETWORK_STATE) para conocer desde la aplicación el estado de red.

3.2.3.6. JSON

JSON ("JSON", 2014) (*JavaScript Object Notation*) es un formato ligero de intercambio de datos. El formato de texto JSON es completamente independiente del lenguaje con el que se está programando pero utiliza convenciones ampliamente utilizadas en distintos lenguajes de programación como C, C++, C#, Java, JavaScript, Perl, Python, etc. Por lo tanto, JSON en un lenguaje ideal para el intercambio de datos.

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un array asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa con arrays, vectores, listas o secuencias.

Estas estructuras están soportadas por todos los lenguajes programación por lo que es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras.

En JSON, dichas estructuras se presentan de las siguientes formas:

- Objeto: conjunto desordenado de pares nombre/valor. Un objeto comienza con "{" y termina con "}". Cada nombre es seguido por ":" y los pares nombre/valor están separados por ",".
- Un array es una colección de valores. Un array comienza con "[" y termina con "]" . Los valores se separan por ",".
- Un valor puede ser una cadena de caracteres con comillas dobles, o un número, o true o false o null, o un objeto o un array. Estas estructuras pueden anidarse.
- Una cadena de caracteres es una colección de cero o más caracteres Unicode, encerrados entre comillas dobles, usando barras divisorias invertidas como escape. Un carácter está representado por una cadena de caracteres de un único carácter. Una cadena de caracteres es parecida a una cadena de caracteres C o Java.
- Un número es similar a un número C o Java, excepto que no se usan los formatos octales y hexadecimales.

Utilización de JSON en el módulo *Registro e identificación de usuario*

La aplicación Android envía consultas al servidor Apache mediante HTTP. Para que la aplicación Android obtenga las respuestas de dicho servidor en formato JSON, los scripts PHP almacenados en el servidor representan la información a devolver a la aplicación Android en formato JSON mediante la función:

- string json_encode (mixed $value): Devuelve un string con la representación JSON de *value*.

La Figura 3.18 muestra la presentación de los gustos de un determinado usuario en formato JSON. Esta respuesta es la que recibe la aplicación Android cuando un usuario se identifica y que contiene los gustos televisivos y de cine registrados en la base de datos MySQL. La aplicación Android debe extraer esta información y almacenarla en la base de datos SQLite. La Tabla 3.9 describe el conjunto de pares nombre/valor que forman la respuesta JSON de la Figura 3.18.

```
{"resultado":1,
 "mensaje":"Gustos registrados!",
 "gustos":[{"usuario":"irmg",
            "deportes":1,
            "cine":1,
            "documentales":1,
            "musica":1,
            "noticias":1,
            "series":1,
            "programas":1,
            "realites":1,
            "concursos":1,
            "genero":"Comedia",
            "pais":"Alemania del Oeste (RFA)",
            "desde":"------",
            "hasta":"------",
            "excluirseries":0,
            "excluirdocumentales":0
           }
          ]
}
```


Figura 3.18: Representación en formato JSON de la respuesta envíada a la aplicación Android al solicitar los gustos de un determinado usuario

Nombre	Valor
resultado	Almacena el entero que representa el estado de la operación de consulta: ■ 1: la consulta se ha llevado a cabo correctamente ■ 0: la consulta no se ha llevado a cabo correctamente
mensaje	Almacena el mensaje que se mostrará al usuario una vez que la operación se ha realizado.
gustos	Almacena el *Array* que contiene en su primera posición un objeto JSON que almacena el conjunto de pares nombre/valor asociados a los gustos del usuario. Estos son: ■ usuario: almacena una cadena con el nombre de usuario ■ deportes: almacena un entero que representa si el usuario quiere obtener recomendaciones de deporte en la televisión (1: Sí, 0: No). ■ cine: almacena el entero que representa si el usuario quiere obtener recomendaciones de cine en la televisión (1: Sí, 0: No). ■ documentales: almacena el entero que representa si el usuario quiere obtener recomendaciones de cine en la televisión (1: Sí, 0: No). ■ musica: almacena el entero que representa si el usuario quiere obtener recomendaciones de música en la televisión (1: Sí, 0: No). ■ noticias: almacena el entero que representa si el usuario quiere obtener recomendaciones de noticias en la televisión (1: Sí, 0: No). ■ series: almacena el entero que representa si el usuario quiere obtener recomendaciones de series en la televisión (1: Sí, 0: No). ■ programas: almacena el entero que representa si el usuario quiere obtener recomendaciones de programas en la televisión (1: Sí, 0: No). ■ realities: almacena el entero que representa si el usuario quiere obtener recomendaciones de realities en la televisión (1: Sí, 0: No). ■ concursos: almacena el entero que representa si el usuario quiere obtener recomendaciones de concursos en la televisión (1: Sí, 0: No). ■ genero: almacena en formato cadena el género de cine sobre el que se desea obtener recomendaciones de películas ■ pais: almacena en formato cadena el país de origen sobre el que se desea obtener recomendaciones de películas ■ desde: almacena en formato cadena la fecha desde la cual se desea obtener resultados en las recomendaciones de cine. ■ hasta: almacena en formato cadena la fecha hasta la cual se desea obtener resultados en las recomendaciones de cine. ■ excluirseries: almacena el entero que representa si el usuario desea excluir series de las recomendaciones de películas (1: Sí, 0: No). ■ excluirdocumentales: almacena el entero que representa si el usuario desea excluir documentales de las recomendaciones de películas (1: Sí, 0: No).

Tabla 3.9: Pares nombre/valor del objeto JSON que representa los gustos del usuario

Por otro lado, la respuesta en formato JSON que recibe la aplicación Android cuando el usuario se registra o sobrescribe sus gustos en la base de datos, únicamente contiene el conjunto de pares

nombre/valor asociados al resultado y al mensaje de la operación efectuada.

La aplicación Android debe extraer la información contenida en la respuesta en formato JSON. Para ello, utiliza las clases org.json.JSONObject ("JSON", 2014) y org.json.JSONArray ("JSON", 2014). La Tabla 3.10 ("JSON", 2014) presenta los métodos pertenecientes a dichas clases que han sido utilizados en la aplicación Android desarrollada.

Método	Clase	Descripción
JSONObject(java.lang.String source)	org.json.JSONObject	Constructor que crea una instancia de la clase JSONObject a partir de la cadena de texto pasada como parámetro. Dicha cadena se obtiene de la respuesta del servidor y comienza con "{" y termina con "}".
JSONObject getJSONObject(java.lang.String key)	org.json.JSONObject	Obtiene el objeto de la clase JSONObject asociado a la clave pasada como parámetro.
JSONArray getJSONArray(java.lang.String key)	org.json.JSONObject	Obtiene el objeto de la clase getJSONArray() asociado a la clave pasada como parámetro.
java.lang.String getString(java.lang.String key)	org.json.JSONObject	Obtiene la cadena asociada a la clave pasada como parámetro.
int getInt(java.lang.String key)	org.json.JSONObject	Obtiene el entero asociado a la clave pasada como parámetro.
JSONObject getJSONObject(int index)	org.json.JSONArray	Obtiene el objeto de la clase JSONObject en la posición *index* del array.

Tabla 3.10: Utilización de las clases org.json.JSONObject y org.json.JSONArray

3.3. Implementación de operaciones generales

Este apartado describe la implementación de las siguientes operaciones generales en Android que han sido necesarias en el desarrollo de la aplicación: ejecución de tareas en segundo plano mediante la utilización de la clase AsyncTask proporcionada por Android., extracción de contenido HTML de una página web a través de la librería JSoup, carga asíncrona de imágenes remotas, e implementación del reconocimiento de voz y de la síntesis de texto a voz con el fin de dotar a la aplicación de un sistema de diálogo.

3.3.1. Ejecución de tareas en segundo plano mediante la utilización de la clase AsyncTask proporcionada por Android

Cualquier operación larga o costosa que se realice en el hilo principal, siendo este el hilo donde se ejecutan todas las operaciones que gestionan la interfaz de usuario de la aplicación, va a bloquear la ejecución del resto de componentes de la aplicación y de la interfaz, produciendo al usuario un efecto evidente de lentitud, bloqueo, o mal funcionamiento en general. Además, dado que Android monitoriza las operaciones realizadas en el hilo principal y detecta aquellas que superen los 5 segundos, mostrará el mensaje de *Application Not Responding* en el que el usuario debe decidir entre forzar el cierre de la aplicación o esperar a que termine. Para evitar este efecto, Android proporciona la clase auxiliar AsyncTask, que permite ejecutar tareas en segundo plano en Android.

La forma básica de utilizar la clase AsyncTask consiste en crear una nueva clase que extienda de ella y sobrescribir varios de sus métodos entre los que se reparte la funcionalidad de la tarea. Estos métodos son los siguientes:

- onPreExecute(): Se ejecuta antes del código principal de la tarea. Se suele utilizar para preparar la ejecución de la tarea, inicializar la interfaz, etc.
- doInBackground(): Contiene el código principal de la tarea.
- onProgressUpdate(): Se ejecuta cada vez que se llame al método publishProgress() desde el método doInBackground().
- onPostExecute(): Se ejecuta cuando finaliza la tarea, o dicho de otra forma, tras la finalización del método doInBackground().
- onCancelled(): Se ejecuta cuando se cancela la ejecución de la tarea antes de su finalización normal.

El método doInBackground() se ejecuta en un hilo secundario (por lo que no se puede interactuar con la interfaz), pero sin embargo todos los demás se ejecutan en el hilo principal, por lo que dentro de ellos se puede hacer referencia directa a los controles de usuario para actualizar la interfaz. Además, el método doInBackground() permite llamar periódicamente al método publishProgress() para que automáticamente desde el método onProgressUpdate() se actualice la interfaz si es necesario. Al extender una nueva clase de AsyncTask se indican tres parámetros:

1. El tipo de datos que se recibe como entrada de la tarea en el método doInBackground().
2. El tipo de datos con el que se actualizará el progreso de la tarea.
3. El tipo de dato*s* que se devuelve como resultado de la tarea, que será el tipo de retorno del método doInBackground() y el tipo del parámetro recibido en el método onPostExecute().

En la aplicación desarrollada para dispositivos móviles Android, se ha ejecutado en segundo plano la operación de enviar peticiones HTTP al servidor Apache con consultas a la base de datos MySQL descrita en la Sección 3.2.3.5 y la operación de extracción de contenido de las páginas web a través de la librería Jsoup descrita en la Sección 3.3.2.

3.3.2. Extracción de contenido de las páginas web a través de la librería JSoup

La extracción de contenido HTML de páginas web es necesario para obtener la información acerca de la programación televisiva, de la cartelera, y de las películas, necesaria en todos los módulos de la aplicación.

JSoup ("JSoup", 2013) es una librería Java que permite al usuario trabajar con HTML. Proporciona una API de gran utilidad para extraer y manipular datos, utilizando métodos DOM, CSS y jQuery.

JSoup implementa la especificación WHATWG HTML5 (Apple Computer, Inc. et al., 2014). JSoup está diseñado para hacer frente a todas las variedades de HTML que se puedan encontrar en una web, y algunas de las características más importantes de esta librería que han propiciado su elección como técnica a la hora de hacer *Web Scraping* son las siguientes:

- Analiza el HTML a partir de una URL, un archivo o cadena.
- Es capaz de encontrar y extraer datos utilizando DOM o CSS.
- Manipula los elementos, atributos y texto de HTML.

Dadas las características de la librería JSoup resulta ideal emplearla para obtener los datos de interés sobre programación de televisión, películas y cartelera requeridos por los módulos principales de la aplicación. La versión actual de dicha librería es la 1.7.3 y se puede descargar como archivo JAR en la propia página de JSoup ("JSoup", 2013). Para poder utilizarla en la aplicación desarrollada se debe incluir el archivo JAR descargado en la carpeta *libs* perteneciente al proyecto que se ha creado en eclipse.

La lógica empleada para realizar el scraping no es compleja, únicamente se basa en seleccionar aquel código HTML que contiene la información que se quiere analizar. En primer lugar, se crea una instancia de la clase Document a partir de la Url de la página web de la cual se desea extraer la información:

```
Document doc = Jsoup.connect("url").get();
```

Para conectarse a la página web de la cual se desea extraer información se va a utilizar la clase AsyncTask, descrita previamente en la Sección 3.3.1 En primer lugar, se crea una barra de progreso dentro del método onPreExecute(). Todas las operaciones correspondientes a la conexión a la página web y extracción de la información del documento HTML mediante la librería JSoup se realizan dentro del método doInBackground(). Una vez que se obtiene la información del código HTML de la página web, dentro del método onPostExecute(), se deshabilita la barra de progreso y se procesa la información obtenida. Para la extraer la información del documento HTML se ha utilizado métodos DOM y CSS.

Métodos CSS

Para realizar las consultas CSS se utiliza el método select(String query) que puede ser invocado por instancias de la clase Document, Element, y Elements. El argumento de dicho método es el selector CSS correspondiente. Dicho método devuelve una lista de elementos (Elements) que proporciona un amplio número de métodos que permiten extraer y manipular datos. Alguno de los selectores CSS utilizados en la aplicación desarrollada se describen en la tabla 3.11.

	Selector	Descripción	Ejemplo
Selectores principales	*	Selecciona todos los elementos	*
	tag	Selecciona aquellos elementos que contienen la etiqueta *tag*	a, div
	#id	Selecciona aquellos elementos cuyo valor del atributo ID es *id*	div#wrap, #logo
	.class	Selecciona aquellos elementos pertenecientes a la clase con nombre *class*	div.left, .result
	[attr]	Selecciona aquellos elementos que contienen el atributo indicado sea cual sea su valor.	a[href], [title]
	[^attrPrefix]	Selecciona aquellos elementos que contienen el atributo que comienza con el prefijo *attrPrefix*.	div[^data-]
	[attr=val]	Selecciona aquellos elementos que contienen el atributo *attr* con valor *val*.	img[width=500]
	[attr^=val], [attr$=val], [attr*=val]	Selecciona aquellos elementos cuyo atributo comienza, termina o contiene el valor indicado por *val*.	a[href^=http:], img[src$=.png] , a[href*=/search/]
Combinación de Selectores	E F	Elemento F que desciende de un elemento E.	div a, .logo h1
	E > F	Elemento F hijo directo de E.	ol > li
	E + F	Elemento F precedido directamente por el elemento del mismo nivel jerárquico E.	li + li, div.head + div
	E ~ F	Elemento F precedido por el elemento del mismo nivel jerárquico E.	h1 ~ p
	E, F, G	Todos los elementos coincidentes E, F, o G.	a[href], div, h3
	:has(selector)	Selecciona aquellos elementos que contienen por lo menos un elemento devuelto por el selector *selector*.	div:has(p)
	:not(selector)	Selecciona aquellos elementos que no contienen el elemento devuelto por el selector *selector*.	div:not(:has(div))

	Selector	Descripción	Ejemplo
Pseudo Selectores	:contains(text)	Selecciona aquellos elementos que contienen el texto especificado por *text*. La búsqueda no es sensible a mayúsculas. El texto puede aparecer en el elementos encontrado o en cualquiera de sus descendientes.	p:contains(jsoup)
	:containsOwn(text)	Selecciona aquellos elementos directos que contienen el texto especificado por *text*. La búsqueda no es sensible a mayúsculas. El texto debe aparecer en el elemento encontrado, sin tener en cuenta sus descendientes.	p:containsOwn(jsoup)

Tabla 3.11: Selectores CSS utilizados en la aplicación

Métodos DOM

La Tabla 3.12 y la Tabla 3.13, describen los métodos DOM de la clase Element y Elements respectivamente, que han sido útiles en el desarrollo de la aplicación.

Método	Descripción
Elements getElementsByTag(String tagName)	Encuentra la lista de elementos que vienen delimitados por la marca (*tag*) pasada como parámetro.
Elements getElementsByAttribute(String key)	Encuentra la lista de elementos que contienen el atributo pasado como parámetro.
Elements getElementsByAttributeValue(String key, String value)	Encuentra la lista de elementos que contienen el atributo pasado como primer parámetro con el valor pasado como segundo parámetro.
Elements siblingElements(), Element firstElementSibling(), Element lastElementSibling(), Element nextElementSibling(), Element previousElementSibling()	Se invocan para devolver y recorrer la lista de elementos del mismo nivel jerárquico que el elemento que los invoca.
Elements children(), Element child(int index)	Se invocan para devolver y recorrer la lista de elementos hijo del elemento que los invoca.
String ownText()	Devuelve el texto perteneciente al elemento que lo invoca. No incluye el texto perteneciente a los hijos de dicho elemento.
public String text()	Devuelve el texto perteneciente al elemento que lo invoca y el de sus hijos.

Tabla 3.12: Métodos DOM de la clase Element

Método	Descripción
String attr(String attributeKey)	Devuelve el valor del atributo del primer elemento que lo contenga.
String text()	Devuelve el texto combinado de todos los elementos de la lista.
boolean isEmpty()	Indica si la lista de elementos está vacía.

Método	Descripción
Element get(int index)	Devuelve el elemento de la lista en la posición correspondiente al parámetro de entrada.
Element first()	Devuelve el primer elemento de la lista.
Element last()	Devuelve el último elemento de la lista.

Tabla 3.13: Métodos DOM de la clase Elements

3.3.3. Carga Asíncrona de Imágenes Remotas

La carga asíncrona de imágenes remotas se utiliza en los módulos ***Búsqueda de películas y Cartelera***. Tal y como se observa en la Figura 3.19, el módulo ***Búsqueda de películas*** utiliza la carga asíncrona de imágenes remotas al mostrar la lista de resultados asociados a la búsqueda (Figura 3.19.a) y al mostrar la película correspondiente al resultado de la búsqueda (Figura 3.19.b). Por otro lado, el módulo ***Cartelera*** utiliza la carga asíncrona de imágenes remotas al mostrar la cartelera correspondiente al cine seleccionado por el usuario, tal y como se observa en la Figura 3.20.

La carga asíncrona de imágenes remotas se realiza en los siguientes pasos:

1. Descarga de las imágenes de una página web y su posterior carga en la aplicación.

2. Almacenamiento de las imágenes descargadas en la caché del dispositivo para su posterior carga.

3. Carga de las imágenes en un elemento de tipo ListView. Este paso se realiza únicamente si se han descargado varias imágenes y se desea que se muestren en una lista tal y como se observa en las Figura 19.a y en la Figura 20.

(a) Resultados asociados a la búsqueda del usuario

(b) Resultado seleccionado por el usuario

Figura 3.19: Capturas de pantalla de la carga asíncrona de imágenes remotas en el módulo Búsqueda de películas

Figura 3.20: Captura de pantalla de la carga asíncrona de imagenes remotas en el móidulo Cartelera

Para realizar la carga asíncrona de imágenes remotas y su posterior almacenamiento en la memoria caché del dispositivo se ha partido del código fuente del sitio web *TechnoTalkative* (Mayani, 2012). El objetivo de dicho código es descargar las imágenes de la página web correspondiente y almacenarlas en la memoria caché del dispositivo, de esta manera, antes de cargar una imagen se comprueba si existe en la memoria local del dispositivo y así evitar su descarga. Las clases necesarias para cargar la imagen y almacenarla en la caché del dispositivo se listan a continuación:

- **ImageLoader.java**: Mediante la utilización de su método DisplayImage(Url, ImageView), se carga y almacena la imagen correspondiente a la Url pasada como parámetro. La imagen se muestra en el elemento de tipo ImageView que se pasa como parámetro de entrada.

- **MemoryCache.java, FileCache.java, Utils.java**: Clases necesarias para cargar de forma asíncrona una imagen remota y almacenarla en la memoria local del dispositivo. La clase FileCache.java, se encarga de la creación de la carpeta *TTImages_cache*, que sirve para almacenar en la memoria caché del dispositivo las imágenes que requiere la aplicación. Además, dicha clase se utiliza para cargar la imagen de la aplicación si esta ya existe en la memoria local del dispositivo. La clase FileCache.java permite además borrar las imágenes almacenadas en dicha carpeta a través de su método clear().

Para poder utilizar este procedimiento de carga asíncrona y almacenamiento de imágenes es necesario incluir los permisos INTERNET y WRITE_EXTERNAL_STORAGE en el fichero AndroidManifest.xml.

El módulo ***Búsqueda de películas*** utiliza este mecanismo para cargar la imagen asociada al resultado de la búsqueda. La carga asíncrona permite al usuario seguir utilizando la aplicación sin la necesidad de esperar a que se cargue la imagen. De no ser posible la carga de la imagen, se muestra una imagen por defecto.

Por otro lado, el módulo ***Cartelera*** y el módulo ***Búsqueda de películas*** muestran respectivamente una lista con las películas en cartelera y con el resultado de la búsqueda de películas. Por lo tanto, las imágenes se deben cargar en un elemento de tipo ListView. Para ello, se ha creado la clase **LazyAdapter.java**, mediante la cual se crea un adaptador para el elemento de tipo ListView. Dicho elemento está formado por diversas filas que contienen elementos de tipo ImageView en los cuales se van mostrando las imágenes descargadas.

3.3.4. Implementación del Reconocimiento de Voz y de la Síntesis de voz

Exceptuando el módulo ***Registro e identificación de usuario***, que únicamente permite la entrada y salida de datos mediante el teclado y pantalla respectivamente, el resto de módulos incorporan el reconocimiento de voz y la síntesis de texto a voz con el fin de dotar al sistema de las características propias de un sistema de diálogo.

Integración del reconocimiento de voz en la aplicación para dispositivos móviles Android

La forma más simple de implementar el reconocimiento de voz en la aplicación Android consiste utilizar cualquier servicio de reconocimiento de voz que el dispositivo haya registrado para recibir un *RecognizerIntent*, siendo únicamente necesario utilizar desde la aplicación las constantes de la clase android.speech.RecognizerIntent descritas en la Sección 2.2.2.2 La aplicación de búsqueda por voz de Google, instalada por defecto en la mayoría de los dispositivos Android, responde a un *RecognizerIntent* mostrando una ventana con el texto "Habla ahora". Posteriormente, transfiere la voz emitida por el usuario a los servidores de Google que se encargan de realizar el reconocimiento. Finalmente, los resultados del reconocimiento de voz son devueltos a la aplicación para que puedan ser procesados.

Para integrar el reconocimiento de voz mediante la clase android.speech.RecognizerIntent se deben seguir los pasos cuya implementación fue ya descrita en la Sección 2.2.2.3: verificar que existe la actividad para reconocer voz, invocar a la actividad de reconocimiento de voz y procesar los resultados. Además, dado que para que funcione el reconocimiento de voz es necesario tener acceso a Internet, en el archivo AndroidManifest.xml se dará dicho permiso (android.permission.INTERNET).

En cuanto a los parámetros de la actividad de reconocimiento de voz únicamente se ha especificado el modelo de lenguaje utilizado a través de la clave EXTRA_LANGUAJE_MODEL y el idioma en el que se realiza el reconocimiento a través de la clase EXTRA_LANGUAJE. Se ha establecido el modelo de lenguaje FREE_FORM. Para definir el idioma en el que se realiza el reconocimiento se ha utilizado un código de idioma IETF, tal como define el estándar BCP 47. Para todos los módulos se ha establecido el español como idioma en el que se realiza el reconocimiento a exceptuando el módulo ***Búsqueda de películas*** que permite al usuario especificar en diferentes idiomas el título de la película que se desea buscar. Las opciones para el idioma del reconocimiento de voz en el módulo ***Búsqueda de películas*** son español, inglés, italiano, francés y alemán.

Integración de la síntesis de texto a voz en la aplicación para dispositivos móviles Android

La forma más simple de implementar la síntesis de texto a voz en la aplicación Android consiste en utilizar cualquier motor de síntesis de voz instalado en el dispositivo, el cual permite integrar de forma sencilla la síntesis de voz a cualquier aplicación mediante la utilización del paquete android.speech.tts perteneciente al SDK de Android, en particular mediante la utilización de su clase android.speech.tts.TextToSpeech. Tal y como se ha explicado, se ha utilizado el motor de síntesis IVONA TTS HQ, ya que ofrece multitud de voces femeninas y masculinas de alta calidad, es actualmente es gratuito y se encuentra disponible en Google Play por lo que se puede descargar de forma sencilla.

Para integrar la síntesis de texto a voz en la aplicación para dispositivos móviles Android se ha seguido el estudio realizado en la Sección 2.3.2. En particular, se han implementado los pasos listados a continuación.

1. Solicitud de los recursos necesarios para llevar a cabo la síntesis de texto a voz e inicialización del motor de síntesis (Sección 2.3.2.2) : Este paso se realiza cuando se pulsa el botón dentro de cada actividad. Posteriormente se realizarán los pasos 2 y 3.

2. Configuración del idioma del motor de síntesis (Sección 2.3.2.3): se ha establecido el español como idioma para la síntesis de texto a voz en todos los módulos del sistema a excepción del módulo ***Búsqueda de películas***, en el que se establece como idioma aquel del país de procedencia de la película para pronunciar el director, el reparto y el título original. Las opciones que se han utilizado en la aplicación para el idioma del sintetizador son español, inglés, portugués, italiano, alemán y francés.

3. Reproducción de una cadena de texto con el sintetizador de voz (Sección 2.3.2.4).

4. Finalizar el motor de síntesis (Sección 2.3.2.8): Se utiliza siempre que se quiera detener el motor de síntesis o bien cuando se pulsa el botón  o cuando se selecciona alguna opción que dirija al usuario a otra actividad.

3.4. Conclusiones

A continuación se resume, en base al estudio realizado en el presente capítulo, los aspectos principales de las tecnologías utilizadas y de la implementación de las operaciones generales y se justifica su utilización en el desarrollo de la aplicación para dispositivos móviles del presente proyecto.

- La elección de la **Plataforma Android** para el desarrollo de la aplicación multimodal para dispositivos móviles del presente proyecto tiene las siguientes ventajas:

  - Al ser un sistema de código abierto cualquiera puede estudiar, modificar, mejorar y distribuir el sistema Android sin ningún tipo de restricción. La comunidad de desarrolladores es muy activa y está creando continuamente soluciones para Android. Además, Android ofrece la opción al desarrollo privado mediante la publicación comercial de aplicaciones, permitiendo a cada desarrollador decidir cómo quiere distribuir su propio trabajo.
  - A pesar de que Google evita utilizar el término demasiado, el que se utilice el **lenguaje de programación Java** para desarrollar las aplicaciones para dispositivos móviles Android ayuda a que cualquier programador que tenga una mínima experiencia con el lenguaje de programación Java pueda comenzar a programar aplicaciones para dispositivos móviles Android sin demasiada dificultad.
  - Cualquier aplicación Android es una combinación de componentes lo que ayuda a modularizar funcionalmente las aplicaciones.
  - Android facilita el diseño de interfaces de usuario ya que incorpora un amplia variedad de elementos y además ofrece la posibilidad de definir la interfaz tanto en el código como a través de documentos XML externos. Declarar el diseño de interfaz de usuario en **XML** en lugar de utilizar código en tiempo de ejecución es útil ya que permite crear diferentes diseños para diferentes tamaños u orientaciones de la pantalla.
  - El acceso a los recursos del dispositivo (Ej. Wi-Fi, servicio de reconocimiento de voz, motor de síntesis, base de datos SQLite, etc.) es una tarea sencilla gracias a las bibliotecas API que Android ofrece en su SDK ya que ayudan a que el desarrollador no tenga que programar a bajo nivel para que una aplicación pueda acceder a los componentes de hardware de los dispositivos.
  - La plataforma Android ofrece un entorno de desarrollo completo que incluye un emulador de dispositivos, herramientas para la depuración, la memoria y perfiles de rendimiento, y un *plug-in* para el **IDE de Eclipse** con lo que se facilita enormemente la tarea de programación en este sistema. Así, pueden crearse proyectos completos para Android, incluyendo el manifiesto y la declaración de recursos externos, sin salir del entorno de desarrollo de Eclipse.

- Las razones por las que se utiliza **SQLite** como sistema de gestión de base de datos en los módulos ***Asistente televisivo*** y ***Recomendación de películas*** son:

  - SQLite es parte del SDK de Android y por tanto no es necesaria su descarga ni incluir librerías adicionales durante el desarrollo de la aplicación.
  - No es necesario disponer de acceso a Internet para poder consultar información de la base de datos SQLite por lo que la aplicación puede funcionar en modo *offline* para estos módulos.
  - SQLite realiza operaciones de manera eficiente y es más rápido que otros sistemas de gestión de base de datos como MySQL.

- Los módulos Asistente televisivo y Recomendación de películas no necesitan almacenar una gran cantidad de datos por lo que SQLite es suficientemente grande para alcanzar los requisitos de almacenamiento de dichos módulos.

Sin embargo, SQLite es una base de datos con funcionalidad limitada a comparación con otras como MySQL y por tanto su utilización se desaconseja en aplicaciones cliente-servidor, en sistemas en los que se desee utilizar bases de datos de tamaño muy grande y en aplicaciones que requieran alta concurrencia.

- Para la implementación del módulo ***Registro e identificación de usuario***, se ha desarrollado una **arquitectura cliente-servidor** para que la aplicación Android acceda a una base de datos MySQL de un servidor web remoto y almacene los datos de usuario de forma que en posteriores interacciones con la aplicación, el usuario pueda iniciar sesión desde cualquier dispositivo mediante su nombre y contraseña y que la aplicación pueda acceder a su información de usuario (gustos televisivos y de cine). Para la parte web se ha utilizado el servidor web gratuito **x10hosting** que se encarga de almacenar los ficheros **PHP** y las bases de datos **MySQL** de la aplicación. Además, para facilitar la administración de las bases de datos se utiliza ***PhpMyAdmin*** ya que se trata de un completo y potente administrador de bases de datos MySQL que se utiliza vía web. En cuanto a la aplicación Android (cliente), se conecta y envía una consulta al servidor mediante **HTTP** y obtiene una respuesta en formato **JSON**, tratándose del formato ideal para el intercambio de datos al ser completamente independiente del lenguaje con el que se está programando.

- Para la **ejecución de tareas en segundo plano**, se ha utilizado la clase AsyncTask proporcionada por Android. Esto es necesario ya que cualquier operación larga o costosa que se realice en el hilo principal, siendo este el hilo donde se ejecutan todas las operaciones que gestionan la interfaz de usuario de la aplicación, va a bloquear la ejecución del resto de componentes de la aplicación y de la interfaz, produciendo al usuario un efecto de lentitud, bloqueo, o mal funcionamiento. En la aplicación desarrollada para dispositivos móviles Android, se ha ejecutado en segundo plano la operación de enviar peticiones HTTP al servidor web con consultas a la base de datos MySQL y la operación de extracción de contenido de las páginas web a través de la librería JSoup.

- Dadas las características estudiadas de la librería Java **JSoup**, resulta ideal emplearla para realizar el *web scraping* y obtener los datos de interés sobre programación de televisión, películas y cartelera, requeridos por los módulos principales de la aplicación. La lógica empleada para realizar el *web scraping* no es compleja, únicamente se basa en seleccionar aquel código HTML que contiene la información que se quiere analizar.

- Para realizar la **carga asíncrona de imágenes remotas** y su posterior almacenamiento en la memoria caché del dispositivo se ha partido de un código cuyo objetivo es descargar las imágenes de la página web correspondiente y almacenarlas en la memoria caché del dispositivo, de esta manera, antes de cargar una imagen se comprueba si existe en la memoria local del dispositivo y así evitar su descarga. La carga asíncrona de imágenes remotas se utiliza en los módulos ***Búsqueda de películas*** y ***Cartelera***.

Capítulo 4

DESCRIPCIÓN DETALLADA DE LOS MÓDULOS DEL SISTEMA

En el presente Capítulo se describe de forma detallada cada uno de los módulos que componen la aplicación para dispositivos móviles Android desarrollada. Para cada módulo se detalla la funcionalidad, la arquitectura y flujo de datos, y ejemplos de posibles escenarios de uso.

4.1. Módulo Registro e identificación de usuario

En su primera interacción con la aplicación Android, el usuario registra su nombre de usuario, contraseña y gustos de cine y televisión. Dicha información se envía a un servidor web y se almacena en una tabla de usuarios registrados.

En posteriores interacciones con el sistema, el usuario podrá iniciar sesión al identificarse a través del nombre de usuario y contraseña registrados. De esta forma, el usuario puede identificarse desde cualquier dispositivo Android y tener acceso a sus gustos televisivos y de cine sin la necesidad de volver a rellenar el formulario. Una vez que el usuario ha sido identificado correctamente, la información sobre sus gustos televisivos y de cine almacenada en la tabla de usuarios registrados de la base de datos MySQL, es enviada a la aplicación móvil para que esta la almacene en una base de datos interna SQlite.

Figura 4.1: Captura de pantalla del módulo Registro e identificación de usuario

La Figura 4.1 muestra la captura de pantalla del módulo ***Registro e identificación de usuario*** en la aplicación. Tal y como se observa, el usuario puede identificarse rellenando los campos de nombre de usuario y contraseña si ya ha sido registrado previamente, o bien acceder al formulario de registro pulsando el botón correspondiente. Además, la aplicación Android dispone también de un modo *offline* que permite al usuario acceder a los servicios sin la necesidad de identificarse. El presente módulo puede dividirse en dos sub-módulos: ***Registro de usuario*** e ***Inicio de sesión***.

4.1.1. Sub-módulo Registro de Usuario

4.1.1.1. Funcionalidad

En su primera interacción con la aplicación Android, el usuario registra el nombre de usuario y contraseña. Además, rellena un formulario con sus gustos televisivos y de cine. Posteriormente, la aplicación Android (cliente) envía dicha información a un servidor web que accede a una base de datos MySQL remota con el objetivo de almacenar los datos de usuario en una tabla de usuarios registrados. Consecuentemente, la tabla de usuarios registrados en la base de datos MySQL almacena la siguiente información de usuario:

- El nombre de usuario y contraseña de los usuarios registrados.
- Las preferencias del usuario en cuanto a la programación televisiva (deportes, cine, documentales, música, noticias, series TV, programas, *realities* y concursos).
- Las preferencias sobre cine del usuario en cuanto a género, país, rango de años entre los cuales se desea obtener resultados, y a si se desea excluir series de televisión o documentales de los resultados.

4.1.1.2. Arquitectura y flujo de datos

La Figura 4.2 muestra la arquitectura cliente-servidor que ha sido utilizada para implementar el sub-módulo ***Registro de usuario***.

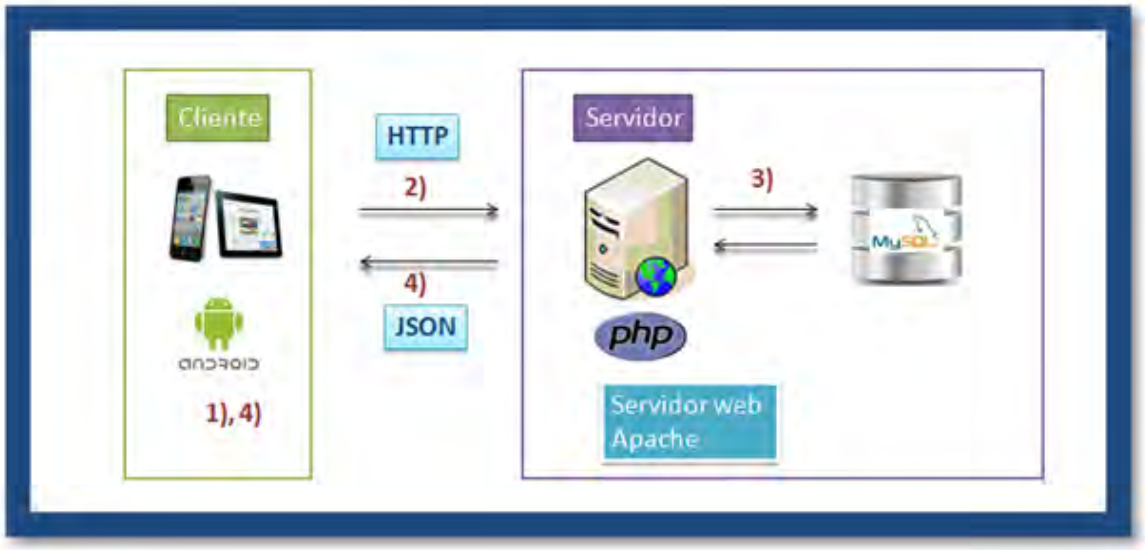


Figura 4.2: Arquitectura cliente-servidor y flujo de datos del sub-módulo Registro de usuario

Tal y como se observa, para la implementación de la parte web se ha utilizado como lenguaje en la parte de servidor PHP (Versión 5.4), como base de datos MySQL y como servidor web Apache. La aplicación Android (cliente) y el servidor PHP se comunican mediante la utilización de JSON y HTTP. El proceso completo se resume en cinco pasos:

1. El usuario registra desde la aplicación Android su nombre de usuario, contraseña y gustos televisivos y de cine en un formulario.
2. La aplicación Android (cliente) envía una petición mediante HTTP al servidor PHP. Dicha petición contiene el nombre de usuario, la contraseña y los gustos televisivos y de cine del usuario.
3. El servidor PHP ejecuta un *script* para acceder a la base de datos MySQL e inserta los datos del usuario en la tabla de usuarios registrados.
4. El servidor PHP envía a la aplicación Android (cliente) información acerca de si la operación de insertar los datos de usuario se ha llevado a cabo con éxito. Dicha información se convierte en formato JSON mediante un script.
5. La aplicación Android lee los datos recibidos y determina si la operación se ha llevado a cabo con éxito.

Para la implementación del presente módulo se ha aplicado la metodología de inserción y manipulación de datos en una base de datos MySQL vista en la Sección 3.2.3.

4.1.1.3. Escenario de uso

Para acceder al formulario de registro el usuario debe pulsar el botón **Registrarse** de la Figura 4.1. La Figura 4.3 muestra una captura de pantalla del formulario de registro de usuario. La Figura 4.3.a muestra la parte del formulario correspondiente al registro del nombre usuario, contraseña y gustos televisivos. La Figura 4.3.b muestra la parte del formulario correspondiente al registro de gustos de cine. Tal y como se observa, el usuario ha rellenado el formulario con su nombre de usuario, contraseña y sus gustos televisivos y de cine. Dicha información se envía a la base de datos MySQL del servidor web remoto al pulsar el botón **Registrar**.

(a) Registro del nombre de usuario, contraseña y gustos televisivos

(b) Registro de gustos de cine

Figura 4.3: Captura de pantalla del formulario del sub-módulo Registro de usuario

La Figura 4.4 muestra una captura de pantalla desde *PhpMyAdmin* de la tabla usuarios del servicio web desarrollado. Se observa que la tabla de usuarios contiene la información registrada por el usuario en el formulario de la Figura 4.3.

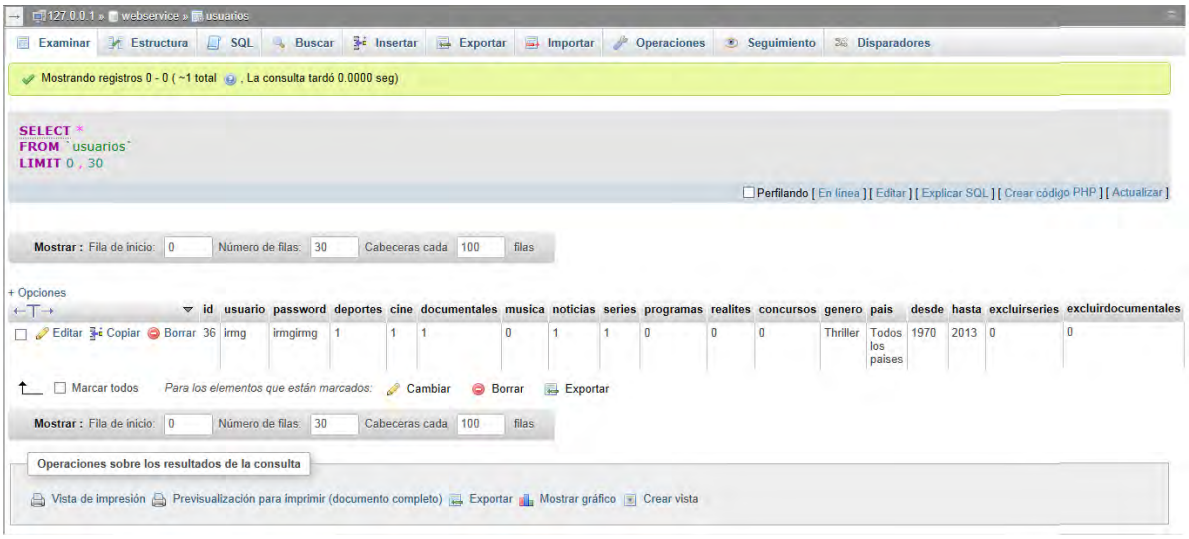


Figura 4.4: Captura desde PhpMyAdmin de la tabla de usuarios de la base de datos MySQL

4.1.2. Sub-módulo Inicio de sesión

4.1.2.1. Funcionalidad

En posteriores interacciones con el sistema, el usuario puede iniciar sesión si se identifica a través del nombre de usuario y contraseña registrados. Una vez que el usuario ha sido identificado correctamente, la información sobre sus gustos televisivos y de cine almacenada en la tabla de usuarios registrados de la base de datos MySQL, es enviada a la aplicación móvil para que esta la almacene en una base de datos interna SQlite. De esta forma, el usuario puede identificarse desde cualquier dispositivo Android y tener acceso a sus gustos televisivos y de cine sin la necesidad de volver a rellenar el formulario. No obstante, la aplicación Android dispone también de un modo *offline* que permite al usuario acceder a los servicios sin la necesidad de identificarse.

Durante el inicio de sesión, se rellenan las siguientes tablas de la base de datos SQLite:

- **Tabla de gustos televisivos**: Almacena las preferencias de un usuario determinado en cuanto a la programación televisiva (deportes, cine, documentales, música, noticias, series TV, programas, realities y concursos). Dicha tabla es accedida desde el módulo que implementa el servicio de asistente televisivo.
- **Tabla de gustos de cine**: Almacena las preferencias sobre cine de un usuario determinado en cuanto a género, país, rango de años entre los cuales se desea obtener resultados, y a si se desea excluir series de televisión o documentales de los resultados.
- **Tabla de películas recomendadas al usuario**: Almacena la lista de películas recomendadas al usuario, extraída del contenido HTML de una página web mediante la utilización de la librería de Java JSoup, de acuerdo a los gustos del usuario registrados en la tabla de gustos de cine. Dicha tabla es accedida desde el módulo que implementa el servicio de recomendación de películas.

4.1.2.2. Arquitectura y flujo de datos

El súbmodulo ***Inicio de sesión*** utiliza la arquitectura cliente-servidor que se muestra en la Figura 4.5. Al igual que en el sobmódulo ***Registro de usuario***, para la implementación de la parte web se utiliza como lenguaje en la parte de servidor PHP (Versión 5.4), como base de datos MySQL y como servidor web Apache. La aplicación Android (cliente) y el servidor PHP se comunican mediante la utilización de JSON y HTTP. Además, la aplicación Android (cliente) accede a la base de datos interna SQLite para almacenar la información sobre los gustos televisivos y de cine. Para obtener la información a almacenar por las tablas de películas recomendadas, la aplicación Android (cliente) se

conecta mediante HTTP al servidor web de la página web que contiene dicha información y la extrae del contenido HTML de la página web mediante la utilización de la librería de Java JSoup.

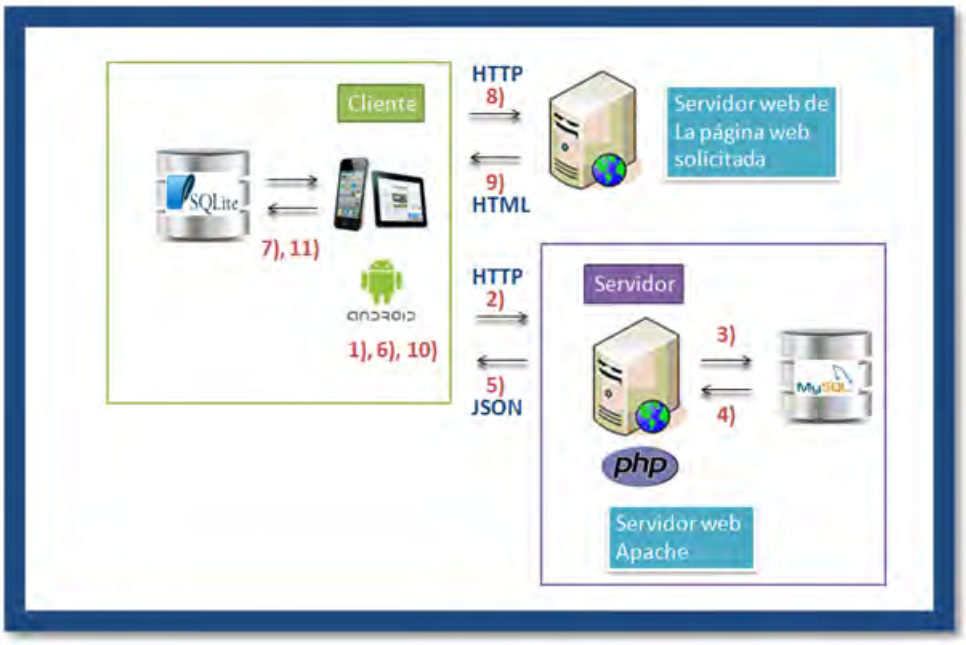


Figura 4.5: Arquitectura y flujo de datos del sub-módulo Inicio de sesión

Tal y como se observa en la Figura 4.5, el proceso completo se resume en los siguientes pasos:

1. El usuario rellena desde la aplicación Android los campos correspondientes al nombre de usuario y contraseña para identificarse en el sistema.

2. La aplicación Android (cliente) envía una consulta mediante HTTP al servidor PHP. Dicha consulta tiene como parámetros el nombre de usuario y la contraseña del usuario que desea identificarse.

3. El servidor PHP ejecuta un *script* para acceder a la base de datos MySQL y realizar la consulta para obtener los datos asociados a los gustos televisivos y de cine solicitados por el cliente. En primer lugar, se debe comprobar en la tabla de usuarios registrados la existencia del nombre de usuario y contraseña enviados en la consulta.

4. La base de datos MySQL devuelve los datos asociados a los gustos televisivos y de cine solicitados al servidor PHP. Dichos datos se obtienen de la fila de la tabla asociada al nombre de usuario y contraseña enviados por el cliente en su consulta.

5. EL servidor PHP analiza los datos devueltos por la base de datos y los convierte en información mediante un script. La información se convierte a formato JSON y se envía a la aplicación Android (cliente).

6. La aplicación Android lee y procesa los datos recibidos.

7. La aplicación Android envía los datos asociados a los gustos televisivos y de cine a la base de datos SQLite y rellena las tablas “Tabla de gustos televisivos” y “Tabla de gustos de cine”.

8. La aplicación Android envía una petición mediante HTTP al servidor del sitio web de *Filmaffinity* y se conecta a la página web que muestra la lista de las mejores películas de acuerdo a los gustos de cine registrados por el usuario.

9. El servidor del sitio web de *Filmaffinity* responde al cliente con el contenido HTML de la página web que muestra la lista de mejores películas de acuerdo a los gustos de cine registrados por el usuario.

10. La aplicación extrae el título y año de dichas películas, que obtiene del contenido HTML de la página web mediante la utilización de la librería JSoup.

11. La aplicación envía dicha información a la base de datos SQLite para rellenar la tabla de películas recomendadas al usuario.

Para la implementación del presente módulo se ha aplicado la metodología de inserción y manipulación de datos en una base de datos SQLite en Android vista en la Sección 3.2.2.2, la metodología de acceso a una base de datos MySQL vista en la Sección 3.2.3 y la metodología para la extracción de contenido de las páginas web a través de la librería JSoup vista en la Sección 3.3.2.

4.1.2.3. Escenario de uso

La Figura 4.6 muestra una captura de pantalla del sub-módulo ***Inicio de sesión***. Tal y como se observa, el usuario ha introducido su nombre de usuario y contraseña. Para que le servidor envíe la información con los gustos televisivos y de cine de dicho usuario a la base de datos SQLite, el usuario debe pulsar el botón **Identificarse** una vez ha introducido su nombre de usuario y contraseña.

Figura 4.6: Captura de pantalla del módulo Inicio de sesión

4.2. Módulo Inicio

4.2.1. Funcionalidad

El módulo ***Inicio*** permite al usuario seleccionar una de las siguientes opciones del menú:

- **Opciones Asistente Televisivo, Recomendación de Películas, Buscar Película y Cartelera**: permite acceder a los módulos ***Asistente televisivo***, ***Recomendación de películas***, ***Buscar película*** y ***Cartelera*** en modo multimodal.

- **Opción Asistente por voz:** permite acceder a los módulos ***Asistente televisivo***, ***Recomendación de películas***, ***Buscar película*** y ***Cartelera*** en modo oral.

- **Opción Registrar Gustos**: permite al usuario acceder al formulario de registro para modificar sus gustos televisivos y de cine a través del módulo ***Registro de gustos televisivos y de cine***.

- **Opción Cerrar Sesión**: permite al usuario cerrar la sesión establecida con la aplicación.

El usuario puede acceder a dichos módulos o cerrar sesión mediante un sistema de diálogo establecido con la aplicación o mediante la selección de la opción correspondiente en el menú que aparece por pantalla.

El módulo ***Inicio*** se inicia cuando el usuario inicia sesión satisfactoriamente desde el módulo ***Registro e inicio de sesión*** o bien al seleccionar la opción para utilizar la aplicación en modo *offline*.

4.2.2. Arquitectura y flujo de datos

La Figura 4.7 muestra la arquitectura y flujo de datos del módulo ***Inicio***. Tal y como se observa, el proceso se resume en los siguientes casos:

1. La aplicación muestra por pantalla un menú que permite al usuario seleccionar el servicio al que desea acceder en modo oral o multimodal, volver a registrar sus gustos o cerrar sesión. Además, la aplicación Android indica al usuario mediante la síntesis de texto a voz que escoja una de las opciones de la lista.
2. El usuario selecciona una de las opciones mediante un sistema de diálogo establecido con la aplicación a través del reconocedor de voz o mediante la selección de la opción correspondiente en el menú que aparece por pantalla.
3. La aplicación procesa la petición del usuario y según la opción solicitada le dirige al servicio correspondiente, al formulario de registro o cierra la sesión.

Figura 4.7: Arquitectura y flujo de datos del módulo Inicio

Para la implementación de este módulo se ha aplicado la metodología descrita en la Sección 3.3.4 para la implementación de reconocimiento de voz y síntesis de texto a voz en aplicaciones para dispositivos móviles Android.

4.2.3. Escenarios de uso

La Figura 4.8 muestra una captura de pantalla de la aplicación Android para el módulo ***Inicio***. Tal y como se observa, se muestra un menú que permite al usuario acceder a los servicios ofrecidos por la aplicación en modo oral o multimodal, volver a registrar sus gustos o cerrar la sesión establecida con la aplicación. Además, si se pulsa el botón , se inicia la síntesis de texto a voz que emite las instrucciones. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz. Al pulsar el botón , el usuario puede utilizar el reconocimiento de voz para indicar a la aplicación que desea hacer: "Registrar gustos", "Asistente televisivo", "Recomendación de películas", "Búsqueda de películas", "Cartelera", o "Cerrar sesión".

Figura 4.8: Captura de pantalla del Modulo Inicio

La Figura 4.9 muestra un ejemplo de cómo se desarrolla el diálogo entre la aplicación y el usuario en el módulo ***Inicio***. "S" indica el diálogo correspondiente al sistema y "U" el diálogo correspondiente al usuario.

S: Bienvenido a la aplicación MyCimeMapp. Debe registrar sus gustos antes de poder utilizar las opciones de Asistente Televisivo y de recomendación de películas. Para ello, diga: Registrar Gustos. Para escuchar las recomendaciones televisivas correspondientes a los gustos registrados diga: Asistente Televisivo. Para escuchar las recomendaciones de películas correspondientes a los gustos registrados diga: Recomendación de películas. Para buscar información de una película concreta diga: Buscar película. Para solicitar la cartelera de su cine más cercano diga: Cartelera. Para salir de la sesión diga: Cerrar sesión. U: Asistente televisivo. (El usuario es dirigido al módulo ***Asistente televisivo***)

Figura 4.9: Diálogo establecido entre el usuario y la aplicación en el módulo Inicio

4.3. Módulo Registro de gustos televisivos y de cine

4.3.1. Funcionalidad

Para poder utilizar los servicios ofrecidos por el módulo ***Asistente televisivo*** y el módulo ***Recomendación de películas***, es necesario que el usuario ingrese sus preferencias en cuanto a la programación televisiva y de cine en un formulario.

Tal y como se ha explicado anteriormente, el registro de gustos televisivos y de cine se realiza por primera vez durante el registro del usuario en el módulo ***Registro e identificación del usuario***. Sin embargo, la aplicación Android permite al usuario volver a rellenar el formulario siempre que lo desee desde el módulo ***Registro de gustos televisivos y de cine***. Una vez que el usuario ha modificado sus gustos a través del formulario, se envía dicha información a la base de datos MySQL remota con el objetivo de actualizar la tabla de usuarios registrados para el usuario correspondiente. Además, la aplicación se encarga de enviar dicha información a la base de datos SQLite con el objetivo de actualizar de la misma forma la tabla de gustos televisivos, la tabla de gustos de cine y la tabla de recomendación de películas.

Tal y como se ha explicado anteriormente, la aplicación dispone de un modo *offline* en el que no es necesaria la identificación del usuario. En este caso, el usuario puede rellenar el formulario con sus gustos televisivos y de cine pero dicha información se envía únicamente a la base de datos interna SQLite.

4.3.2. Arquitectura y flujo de datos

La Figura 4.10 muestra la arquitectura del módulo ***Registro de gustos televisivos y de cine***. Tal y como se observa, el acceso a la base de datos SQLite se realiza desde el lado del cliente. Para obtener la información a almacenar por la tabla de películas recomendadas de la base de datos SQLite, la aplicación Android (cliente) se conecta mediante HTTP al servidor web de la página web que contiene dicha información y la extrae del contenido HTML de la página web mediante la utilización de la librería de Java JSoup. Además, la aplicación Android envía la información sobre los gustos televisivos y de cine al servidor PHP con el objetivo de actualizar la tabla de usuarios registrados de la base de datos MySQL remota.

El proceso se resume en los siguientes pasos:

1. El usuario registra los gustos televisivos y de cine en el formulario.

2. La aplicación Android envía los datos asociados a los gustos televisivos y de cine a la base de datos SQLite y rellena las tablas de gustos televisivos y de gustos de cine.

3. La aplicación Android envía una petición mediante HTTP al servidor del sitio web de *Filmaffinity* y se conecta a la página web que muestra la lista de las mejores películas de acuerdo a los gustos de cine registrados por el usuario.

4. El servidor del sitio web de *Filmaffinity* responde al cliente con el contenido HTML de la página web que muestra la lista de mejores películas de acuerdo a los gustos de cine registrados por el usuario.

5. La aplicación extrae el título y año de dichas películas, que obtiene del contenido HTML de la página web mediante la utilización de la librería JSoup.

6. La aplicación envía dicha información a la base de datos SQLite para rellenar la tabla de películas recomendadas al usuario.

7. La aplicación Android (cliente) envía una petición mediante HTTP al servidor PHP. Dicha petición contiene el nombre de usuario y los gustos televisivos y de cine.

8. El servidor PHP ejecuta un *script* para acceder a la base de datos MySQL y edita la fila correspondiente al nombre de usuario con los nuevos gustos registrados en el formulario.

9. El servidor PHP envía a la aplicación Android (cliente) información acerca de si la operación de modificar los datos de usuario se ha llevado a cabo con éxito. Dicha información se convierte en formato JSON mediante un script.

10. La aplicación Android lee y los datos recibidos y determina si la operación se ha llevado a cabo con éxito.

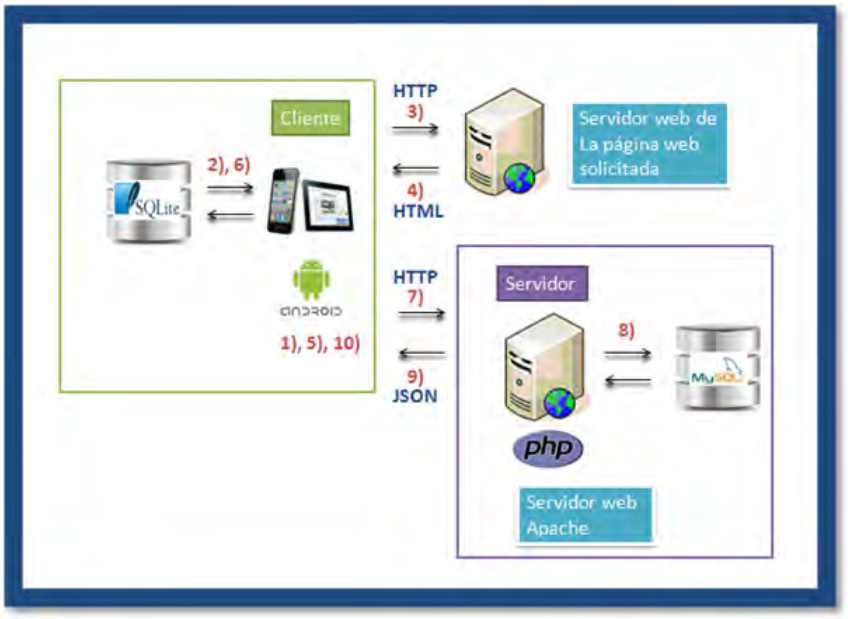


Figura 4.10: Arquitectura y flujo de datos del módulo Registro de gustos televisivos y de cine

Para la implementación del presente módulo se ha aplicado la metodología de inserción y manipulación de datos en una base de datos SQLite en Android vista en la Sección 3.2.2.2, la metodología de inserción y manipulación de datos en una base de datos MySQL vista en la Sección 3.2.3 y la metodología para la extracción de contenido de las páginas web a través de la librería JSoup vista en la Sección 3.3.2.

4.3.3. Escenarios de uso

El módulo ***Registro de gustos televisivos y de cine*** se accede desde el menú del modulo ***Inicio*** mostrado en la Figura 4.8. El usuario puede o bien pulsar el botón de la lista correspondiente al registro de gustos o bien decir la palabra clave "Registrar Gustos" al reconocedor de voz pulsando el botón  .

La Figura 4.11 muestra una captura del formulario de registro de gustos de cine y televisivos. La Figura 4.11.a muestra la parte del formulario correspondiente al registro de gustos televisivos y la Figura 4.11.b muestra la parte del formulario correspondiente al registro de gustos de cine. Dicha información se envía a la base de datos MySQL del servidor web remoto y a la base de datos interna SQLite al pulsar el botón "Registrar" para editar los gustos que el usuario registró previamente. Tal y como se observa, el usuario ha modificado los gustos registrados respecto a los de la Figura 4.3.

La Figura 4.12 muestra una captura de pantalla desde *PhpMyAdmin* de la tabla usuarios del servicio web desarrollado. Se observa que la tabla de usuarios contiene la información ahora registrada por el usuario según la Figura 4.11.

(a) Registro de gustos televisivos (b) Registro de gustos de cine

Figura 4.11: Captura de pantalla del formulario del módulo Registro de gustos televisivos y de cine

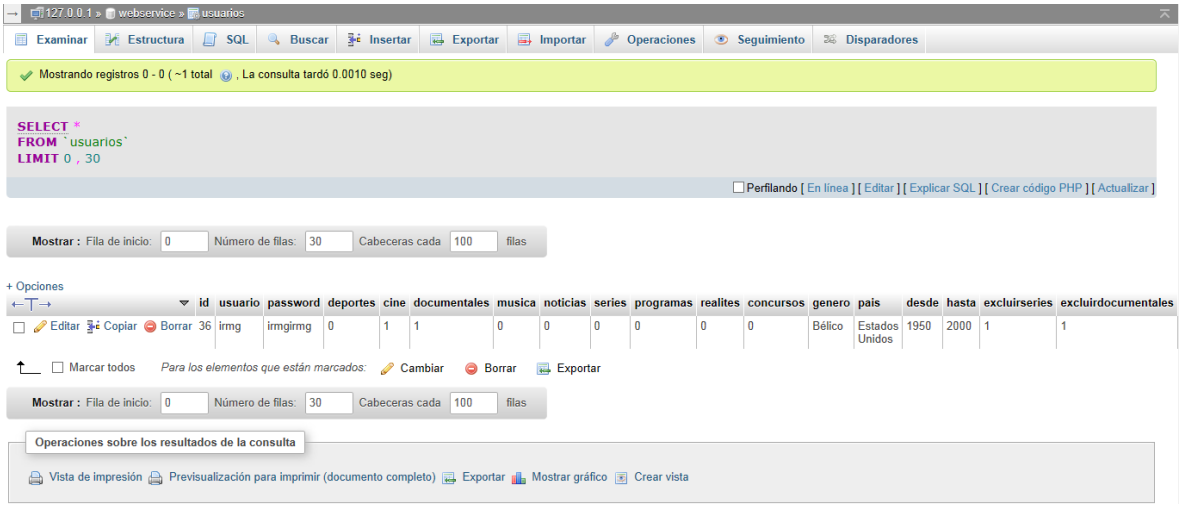


Figura 4.12: Captura desde PhpMyAdmin de la modificación en la base de datos MySQL de los gustos televisivos y de cine

4.4. Módulo Asistente por voz

4.4.1. Funcionalidad

El módulo ***Asistente por voz*** conduce al usuario a los módulos ***Búsqueda de películas***, ***Asistente televisivo***, ***Recomendación de películas***, y ***Cartelera*** en modo oral en función de las decisiones que vaya tomando en cada diálogo. Este módulo utiliza únicamente el reconocedor de voz como entrada al sistema y la síntesis de texto a voz como salida al sistema. La utilización de los módulos ***Búsqueda de películas***, ***Asistente televisivo***, ***Recomendación de películas***, y ***Cartelera*** en modo oral viene descrita, junto a su utilización en modo multimodal, en las Secciones 4.5, 4.6, 4.7 y 4.8 respectivamente. El módulo ***Asistente por voz*** se inicia desde el menú del módulo ***Inicio*** mostrado en la Figura 4.8. El usuario puede o bien pulsar la opción del menú **Asistente por voz** o bien decir la palabra clave "Asistente por voz" al reconocedor de voz pulsando el botón .

4.4.2. Arquitectura y flujo de datos

La Figura 4.13 muestra la arquitectura y flujo de datos del módulo ***Asistente por voz***. El proceso se resume en los siguientes casos:

1. La aplicación indica al usuario mediante la síntesis de texto a voz que escoja el módulo al que desea acceder (***Búsqueda de películas***, ***Asistente televisivo***, ***Recomendación de películas***, o ***Cartelera***).

2. El usuario selecciona una de las opciones mediante un sistema de diálogo establecido con la aplicación a través del reconocedor de voz.

3. La aplicación procesa la petición del usuario y según la opción solicitada le dirige al usuario al módulo correspondiente en modo oral.

Figura 4.13: Arquitectura y flujo de datos del módulo Asistente por voz

Para la implementación de este módulo se ha aplicado la metodología descrita en la Sección 3.3.4 para la implementación de reconocimiento de voz y síntesis de texto a voz en aplicaciones para dispositivos móviles Android.

Las arquitecturas para los módulos ***Búsqueda de películas***, ***Asistente televisivo***, ***Recomendación de películas***, y ***Cartelera*** en modo oral se describen en las Secciones 4.5, 4.6, 4.7 y 4.8 respectivamente.

4.4.3. Escenarios de uso

La Figura 4.14 muestra un ejemplo de cómo se desarrolla el diálogo entre la aplicación y el usuario en el módulo ***Asistente por voz***.

S: ¿Qué Servicio desea solicitar? Para escuchar las recomendaciones televisivas correspondientes a los gustos registrados diga: Asistente Televisivo. Para escuchar las recomendaciones de películas correspondientes a los gustos registrados diga: Recomendación de películas. Para buscar una película diga: Buscar película. Para solicitar la cartelera de su cine preferido diga: Cartelera.
U: Recomendación de películas. (El usuario es dirigido al módulo ***Recomendación de películas***)

Figura 4.14: Diálogo establecido entre el usuario y la aplicación en el módulo Asistente por voz

4.5. Módulo Búsqueda de películas

4.5.1. Funcionalidad

El módulo ***Búsqueda de películas*** permite al usuario solicitar información sobre una película (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios). Dicha información se obtiene de forma dinámica del sitio web de *FilmAffinity* por lo que es necesario tener acceso a Internet para poder utilizar el servicio.

El usuario puede acceder a este módulo interactuando con el sistema de varias formas:

- **Interacción oral:** Mediante la opción del menú **Asistente por voz** del menú principal, que conduce al usuario al módulo ***Búsqueda de películas*** si en su primera decisión el usuario dice al reconocedor la palabra "Buscar película", "Buscar películas" o "Busqueda de películas".
- **Interacción multimodal:** El módulo ***Búsqueda de películas*** en modo multimodal, se accede desde el menú del modulo ***Inicio*** mostrado en la Figura 4.8. El usuario puede o bien pulsar la opción del menú **Búsqueda de películas** o bien decir la palabra clave "Búsqueda de películas" al reconocedor de voz pulsando el botón . En todo momento, la aplicación permite al usuario seleccionar la modalidad con la que desea interactuar con el dispositivo de acuerdo a sus preferencias y a las condiciones del ambiente de uso. El usuario puede interactuar con el sistema mediante el modo táctil, el modo oral, o combinar ambos modos.

4.5.2. Arquitectura y flujo de datos

La Figura 4.15 muestra la arquitectura del módulo ***Búsqueda de películas***. Los pasos correspondientes al flujo de datos en la interacción multimodal se muestran en azul y los correspondientes a la interacción oral se muestran en verde.

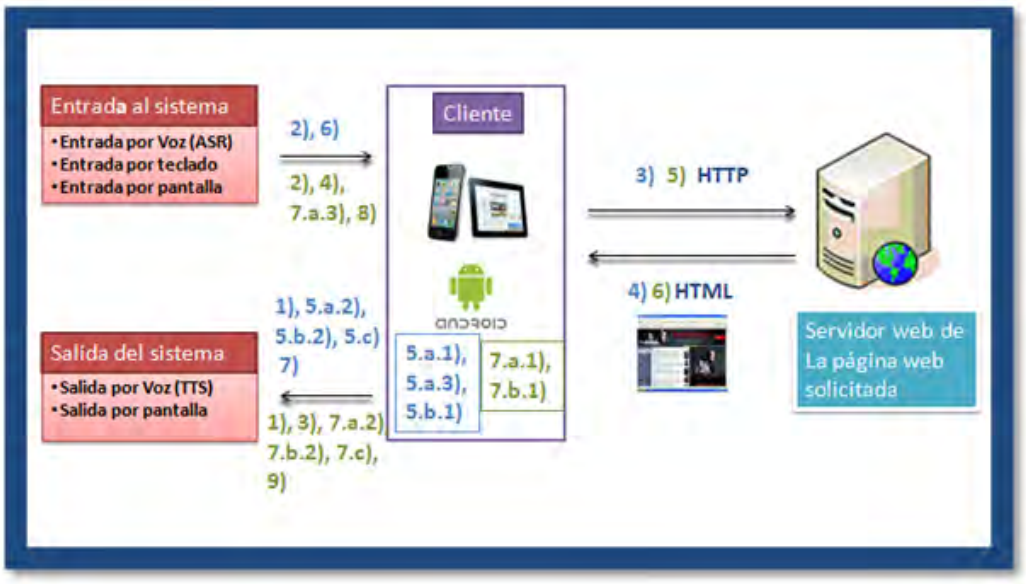


Figura 4.15: Arquitectura y flujo de datos del módulo Búsqueda de películas

Interacción Multimodal

El proceso llevado a cabo cuando se solicita acceder a este módulo mediante interacción multimodal se resume en los pasos listados a continuación:

1. La aplicación Android indica las instrucciones de uso del servicio de ***Búsqueda de películas*** al usuario mediante la síntesis de texto a voz.

2. El usuario comunica a la aplicación, a través del teclado o a través del reconocimiento de voz, el título de la película sobre la que desea obtener información. El usuario puede escoger el idioma del reconocedor de voz por lo que puede pronunciar el título de la película en el idioma que desee.

3. La aplicación Android (cliente) envía una petición mediante HTTP al servidor del sitio web de *FilmAffinity* y se conecta a la página web de la película que solicita el usuario.

4. El servidor del sitio web de *Filmaffinity* responde al cliente con el contenido HTML de la página web de la película solicitada.

5. Llegados a este paso, se pueden dar tres casos:

   a) Se ha encontrado más de un resultado que coincide con el título de la película solicitada por el usuario en cuyo caso se ejecutan los siguientes pasos:

      1) Se muestra la lista con los resultados encontrados por pantalla. La información de dicha lista se extrae, mediante la utilización de la librería JSoup, del contenido HTML de la página web devuelta por el servidor de la página web en el paso 4. Para mostrar las imágenes correspondientes a las películas encontradas, se utiliza el proceso descrito en la Sección 3.3.3 para la carga asíncrona de imágenes remotas y su almacenamiento en la caché del dispositivo.
      2) La aplicación Android indica al usuario, mediante la síntesis de texto a voz, que escoja una de las películas de la lista.
      3) El usuario escoge a través de la interfaz gráfica el elemento de la lista que se corresponda con la película sobre la que desea saber información. Se vuelven a ejecutar los pasos 3 y 4 y en esta ocasión se dará el caso 5.b.

   b) Se ha encontrado un único resultado que coincide con el título de la película solicitada por el usuario en cuyo caso se se ejecutan los siguientes pasos:

      1) La aplicación descarga la imagen la película encontrada utilizando el proceso descrito en la Sección 3.3.3 para la carga asíncrona de imágenes remotas y su almacenamiento en la caché del dispositivo. A continuación, la aplicación extrae, mediante la utilización de la librería JSoup, la información de la película solicitada por el usuario (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios), obtenida del contenido HTML de su página web en el paso 4.
      2) La aplicación pide al usuario, mediante la síntesis de texto a voz, que escoja la información que desea conocer acerca de la película encontrada. Se ejecuta el paso 6.

   c) No se ha encontrado ningún resultado que coincida con el título de la película solicitada por el usuario en cuyo caso se le indica al usuario mediante el sintetizador de voz y se regresa al paso 1.

6. El usuario comunica a la aplicación, a través de la interfaz gráfica o a través del reconocimiento de voz, la información que desea solicitar acerca de la película (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios).

7. La aplicación devuelve la información solicitada por el usuario por pantalla o por voz mediante la síntesis de texto a voz.

Interacción Oral

El proceso llevado a cabo cuando se solicita acceder a este módulo mediante interacción oral se resume en los pasos listados a continuación:

1. La aplicación Android indica las instrucciones de uso del servicio de ***Búsqueda de películas*** al usuario mediante la síntesis de texto a voz.

2. El usuario comunica a la aplicación, a través del reconocimiento de voz, el idioma en el que pronunciará el título de la película sobre la que desea obtener información.

3. La aplicación Android solicita al usuario, mediante la síntesis de texto a voz, que le comunique el título de la película sobre la que desea obtener información.

4. El usuario comunica a la aplicación, a través del reconocimiento de voz, el título de la película sobre la que desea obtener información en el idioma deseado.

5. La aplicación Android (cliente) envía una petición mediante HTTP al servidor del sitio web de *FilmAffinity* y se conecta a la página web de la película que solicita el usuario.

6. El servidor del sitio web de *Filmaffinity* responde al cliente con el contenido HTML de la página web de la película solicitada.

7. Llegados a este paso, se pueden dar tres casos:

   a) Se ha encontrado más de un resultado que coincide con el título de la película solicitada por el usuario en cuyo caso se ejecutan los siguientes pasos:

      1) La aplicación extrae la lista con los resultados encontrados, mediante la utilización de la librería JSoup, del contenido HTML de la página web devuelta por el servidor de la página web en el paso 5.
      2) La aplicación Android indica al usuario mediante la síntesis de texto a voz que escoja una de las películas de la lista de resultados. La aplicación asocia un número a cada elemento de la lista y pide al usuario, mediante la síntesis de texto a voz, que le diga al reconocedor de voz el número asociado a la película sobre la que desea solicitar información.
      3) El usuario le dice a la aplicación Android a través del reconocimiento de voz el número asociado a la película sobre la que desea saber información. Después de esto habrá un único resultado por lo que se ejecutará el paso 6.b.

   b) Se ha encontrado un único resultado que coincide con el título de la película solicitada por el usuario en cuyo caso se ejecutan los siguientes pasos:

      1) La aplicación extrae, mediante la utilización de la librería JSoup, la información de la película solicitada por el usuario (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios), obtenida del contenido HTML de su página web en el paso 5.
      2) La aplicación pide al usuario mediante la síntesis de texto a voz que escoja la información que desea conocer acerca de la película encontrada. Se ejecuta el paso 6.

   c) No se ha encontrado ningún resultado que coincida con el título de la película solicitada por el usuario en cuyo caso se le indica al usuario mediante el sintetizador de voz y se regresa al paso 1.

8. El usuario comunica a la aplicación a través del reconocimiento de voz la información que desea solicitar acerca de la película (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios).

9. La aplicación devuelve la información solicitada por el usuario mediante la síntesis de texto a voz.

Para la implementación de este módulo se han aplicado la extracción de contenido de las páginas web a través de la librería JSoup descrita en la Sección 3.3.2, la carga asíncrona de imágenes remotas descrita en la Sección 3.3.3 y la implementación de reconocimiento de voz y síntesis de texto a voz en aplicaciones para dispositivos móviles Android descrita en la Sección 3.3.4.

4.5.3. Escenarios de uso

La Figura 4.16 muestra, a través de capturas de pantalla sacadas de la aplicación Android, un ejemplo de utilización en modo multimodal del servicio del módulo ***Búsqueda de películas***. En el caso de dicho ejemplo, se encuentra un único resultado para la petición.

A continuación, se describe cada una de las capturas de pantalla que se van mostrando de manera secuencial al usuario durante la utilización del servicio:

1. Menú que permite al usuario indicar a la aplicación el título de la película del que desea solicitar información. El usuario puede escribir el título de la película mediante el teclado o utilizar el servicio de reconocimiento de voz seleccionando el idioma adecuado de la lista que se muestra. Además, si se pulsa el botón , se inicia la síntesis de texto a voz que emite las instrucciones. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz.

2. El usuario indica a la aplicación el título de la película sobre la que desea solicitar información o bien mediante el teclado o mediante el reconocimiento de voz. En el caso del ejemplo el título de la película es *American Beauty*.

   a) El usuario dice el título de la película al reconocedor de voz. Al ser una película cuyo título está en inglés se selecciona dicho idioma para el reconocimiento de voz.

   b) El usuario introduce mediante el teclado el título de la película.

3. Resultado de la película solicitada. Si se pulsa el botón se inicia la actividad de síntesis de texto a voz que informa al usuario sobre el título de la película solicitada y le pide que solicite la información que desea saber acerca de dicha película. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz. El usuario puede seleccionar la opción o bien mediante el reconocimiento de voz pulsando el botón , o bien mediante el teclado seleccionando una de las opciones que se ofrecen.

4. El usuario indica a la aplicación la información que desea solicitar acerca de la película encontrada. Las opciones son: sinopsis, título original, país, duración, director, reparto, género, año, calificación, premios.

   a) El usuario dice al reconocedor de voz la información que desea solicitar acerca de la película encontrada.

   b) El usuario selecciona de la lista la información que desea solicitar acerca de la película encontrada.

5. Información solicitada por el usuario de la película encontrada. En el caso del ejemplo, el usuario a solicitado la sinopsis de la película *American Beauty*. Al pulsarse el botón se inicia la lectura de la sinopsis. En cualquier momento se puede pulsar el botón y se detiene la síntesis de texto a voz.

La Figura 4.17 muestra, a través de una serie de capturas de pantalla sacadas de la aplicación Android, un segundo ejemplo de utilización en **modo multimodal** del servició del módulo ***Búsqueda de películas***. En el caso de dicho ejemplo, se encuentran varios resultados para la petición.

A continuación, se describe cada una de las capturas de pantalla que se muestran de manera secuencial al usuario durante la utilización del servicio:

1. Menú que permite al usuario indicar a la aplicación el título de la película del que desea solicitar información. El usuario puede escribir el título de la película mediante el teclado o utilizar el servicio de reconocimiento de voz seleccionando el idioma adecuado de la lista que se muestra. Además, si se pulsa el botón , se inicia la síntesis de texto a voz que emite las instrucciones. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz.

2. El usuario indica a la aplicación el título de la película sobre la que desea solicitar información o bien mediante el teclado o mediante el reconocimiento de voz. En el caso del ejemplo el título de la película es *American Pie*.

   a) El usuario dice el título de la película al reconocedor de voz. Al ser una película cuyo título está en inglés se selecciona dicho idioma para el reconocimiento de voz.

   b) El usuario introduce mediante el teclado el título de la película.

3. Lista de las películas encontradas para la película solicitada. El usuario selecciona la película de la lista sobre la que desea solicitar información. En el caso del ejemplo selecciona la película *American Pie: El reencuentro*. Además, si se pulsa el botón , se inicia la síntesis de texto a voz que emite las instrucciones y los resultados de la lista. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz.

4. Resultado de la película solicitada. Si se pulsa el botón se inicia la actividad de síntesis de texto a voz que informa al usuario sobre el título de la película solicitada y le pide que solicite la información que desea saber acerca de dicha película. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz. El usuario puede seleccionar la opción o bien mediante el reconocimiento de voz pulsando el botón , o bien mediante el teclado seleccionando una de las opciones que se ofrecen.

5. El usuario indica a la aplicación la información que desea solicitar acerca de la película encontrada. Las opciones son: sinopsis, título original, país, duración, director, reparto, género, año, calificación, premios.

   a) El usuario dice al reconocedor de voz la información que desea solicitar acerca de la película encontrada.

   b) El usuario selecciona de la lista la información que desea solicitar acerca de la película encontrada.

6. Información solicitada por el usuario de la película encontrada. En el caso del ejemplo el usuario a solicitado el reparto de la película *American Pie: El reencuentro*. Al pulsarse el botón se inicia la lectura del reparto. Al detectar que se trata de una película americana el reparto se dirá con acento inglés. En cualquier momento se puede pulsar el botón y se detiene la síntesis de texto a voz.

La Figura 4.18 muestra cómo se desarrolla el diálogo entre el usuario y la aplicación en el ***modo oral*** para el ejemplo de la Figura 4.17.

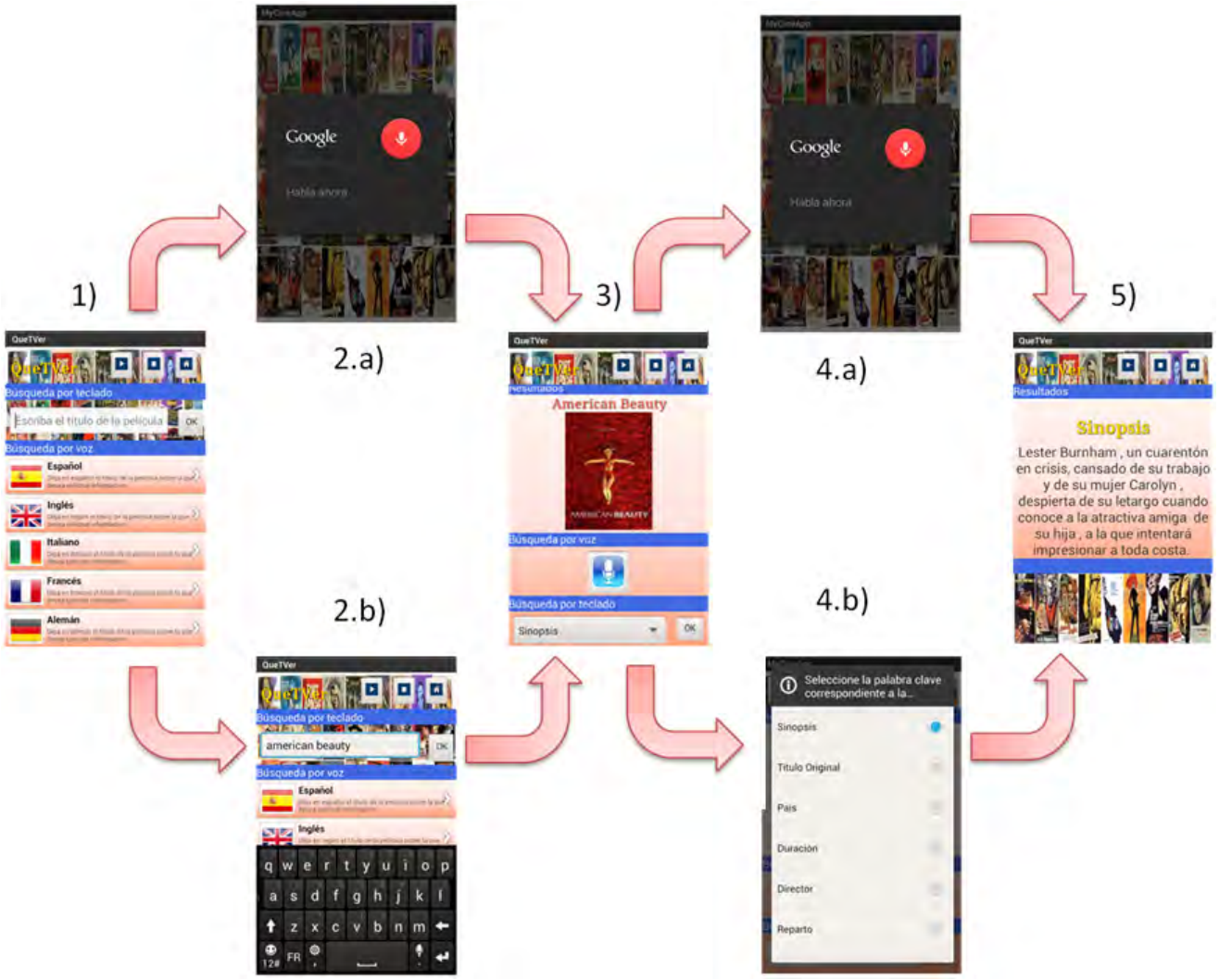


Figura 4.16: Ejemplo del módulo Búsqueda de películas en modo multimodal cuando se encuentra un resultado

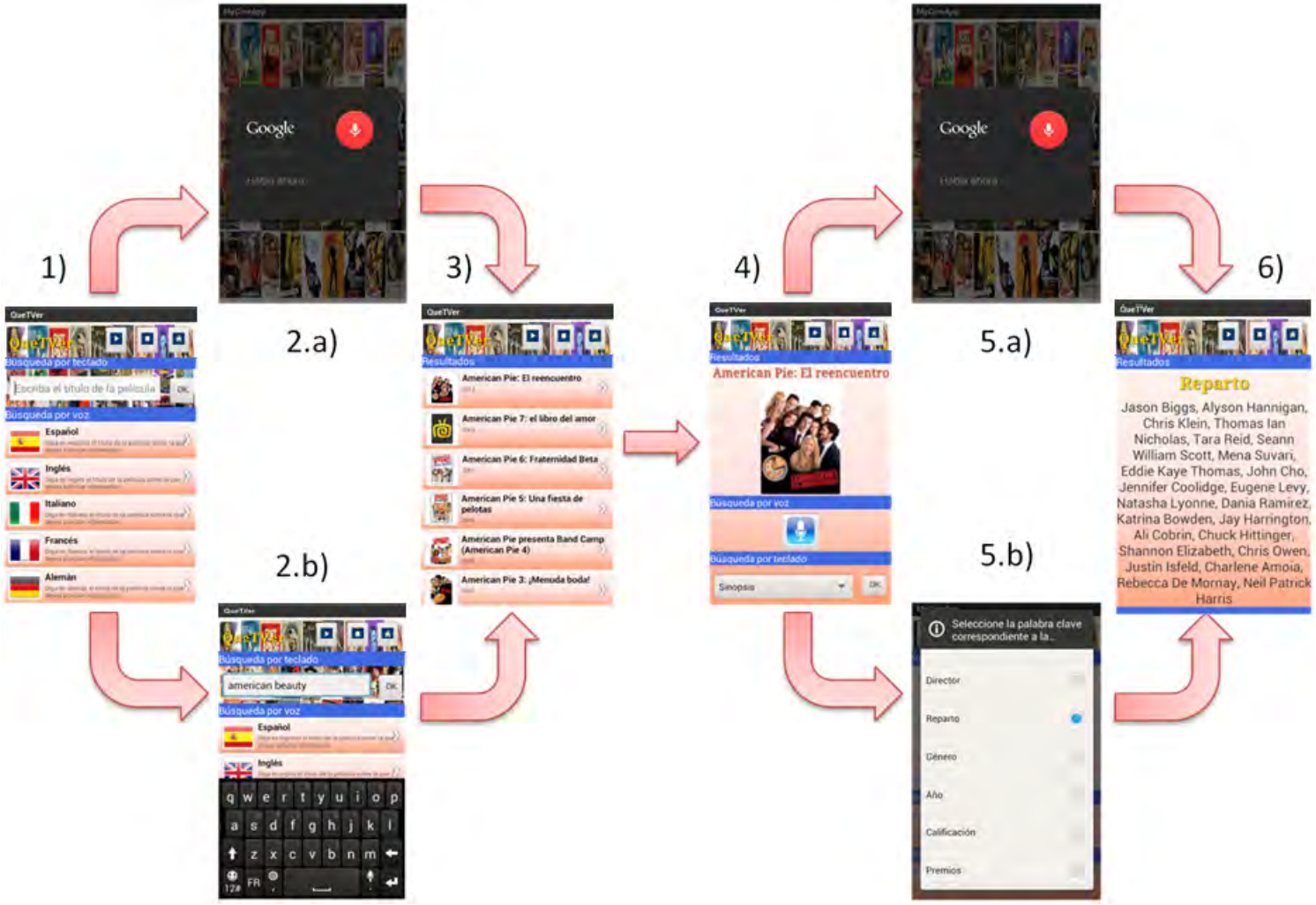


Figura 4.17: Ejemplo del módulo Búsqueda de películas en modo multimodal cuando se encuentra varios resultados

S: ¿Qué Servicio desea solicitar? Para escuchar las recomendaciones televisivas correspondientes a los gustos registrados diga: Asistente Televisivo. Para escuchar las recomendaciones de películas correspondientes a los gustos registrados diga: Recomendación de películas. Para buscar una película diga: Buscar película. Para solicitar la cartelera de su cine preferido diga: Cartelera.

U: Búsqueda de películas.

(El usuario es dirigido al módulo ***Búsqueda de películas***)

S: Diga al reconocedor el idioma en el que pronunciara el titulo de la pelicula sobre la que desea saber informacion: español, ingles, italiano, frances o aleman.

U: Inglés.

S: Diga al reconocedor el título de la película sobre la que desea solicitar información.

U: American Pie

S: Se han encontrado 9 resultados. Para solicitar información sobre la película *American pie*: el reencuentro (año 2012) diga 1, Para solicitar información sobre la película *American pie* 7: el libro del amor (año 2009) diga 2, Para solicitar información sobre la película *American pie* 6: Fraternidad Beta (año 2007) diga 3, ...

U: 3.

S: Para obtener la sinopsis de la película diga: Sinopsis. Para obtener el título original de la película diga: Original. Para obtener el país de la película diga: País. Para obtener la duración de la película diga: Duración. Para obtener el director de la película diga: Director. Para obtener el reparto de la película diga: Reparto. Para obtener el género de la película diga: Género. Para obtener el año de la película diga: Año. Para obtener los premios de la película diga: Premios. Para obtener la calificación de la película diga: Calificación.

U: Sinopsis.

S: Erik, Ryan, y Cooze comienzan la universidad e ingresan en la Fraternidad Beta, presidida por el legendario Dwight Stifler. Pero el problema comienza cuando una fraternidad de chiflados amenaza con poner fin al libertinaje y la Fraternidad Beta tiene que reivindicar su derecho a la fiesta.

Figura 4.18: Diálogo establecido entre el usuario y la aplicación en el módulo Búsqueda de películas en el modo oral

4.6. Módulo Asistente televisivo

Recomienda al usuario la programación diaria de acuerdo con los gustos registrados. El usuario debe actualizar diariamente una base de datos interna, implementada con SQLite, que recoge en varias tablas, una por cada canal de televisión, la programación emitida el día de la actualización. Una vez actualizadas las tablas de programación televisiva, el asistente televisivo permite al usuario seleccionar un canal de televisión determinado y obtener recomendaciones, de acuerdo a los gustos registrados, sobre la programación televisiva emitida el día de la última actualización. La Figura 4.19 muestra una captura de pantalla del menú inicial del módulo ***Asistente televisivo*** en **modo multimodal**.

Figura 4.19: Captura de pantalla del menú inicial del módulo Asistente televisivo

El usuario puede acceder a este módulo interactuando con el sistema de varias formas:

- **Interacción oral:** Mediante la opción del menú **Asistente por voz** del menú principal, que conduce al usuario al módulo ***Asistente televisivo*** si en su primera decisión el usuario dice al reconocedor la palabra "Asistente televisivo".
- **Interacción multimodal:** El módulo ***Asistente televisivo*** en modo multimodal, se accede desde el menú del modulo ***Inicio*** mostrado en la Figura 4.8. El usuario puede o bien pulsar la

opción del menú **Asistente televisivo** o bien decir la palabra clave "Asistente televisivo" al reconocedor de voz pulsando el botón  . En todo momento, la aplicación permite al usuario seleccionar la modalidad con la que desea interactuar con el dispositivo de acuerdo a sus preferencias y a las condiciones del ambiente de uso. El usuario puede interactuar con el sistema mediante el modo táctil, el modo oral, o combinar ambos modos.

El módulo ***Asistente televisivo*** puede dividirse en dos sub-módulos: sub-módulo ***Actualización de la programación televisiva*** y sub-módulo ***Recomendación de programación televisiva***.

El módulo ***Asistente televisivo*** en **modo multimodal** ofrece gran parte de su funcionalidad en modo *offline*, siendo únicamente necesario disponer de acceso Internet cuando se vaya a actualizar la base de datos o si se quiere utilizar el reconocimiento de voz como entrada al sistema. Además, el usuario puede consultar información más detallada de cada película y serie de TV recomendada por el asistente.

Por otro lado, el módulo ***Asistente televisivo*** en **modo oral** necesita acceso a Internet ya que requiere utilizar el reconocimiento de voz en todo momento y además no permite al usuario solicitar información más detallada sobre la película recomendada.

Para poder utilizar el servicio ofrecido por el módulo ***Asistente televisivo*** es necesario haber registrado los gustos de televisión en el cuestionario de registro.

4.6.1. Sub-módulo Actualización de la programación televisiva

4.6.1.1. Funcionalidad

El sub-módulo ***Actualización de la programación televisiva*** se conecta al servidor de la página web *teletexto.com* y solicita información acerca de la programación televisiva de los canales TVE 1, TVE 2, Antena 3, Cuatro, Tele 5, Sexta, TeleMadrid. Además, se conecta a los servidores de las páginas web de la Sexta 3 y *Paramount Channel* para solicitar de la misma forma su programación televisiva. El servidor de dichas páginas responde al cliente con la programación televisiva de los canales solicitados. El cliente procesa la información y la almacena en las tablas de programación televisiva de la base de datos SQLite:

- **Tablas de programación televisiva**: Almacena la información sobre la programación televisiva (TVE 1, TVE 2, Antena 3, Cuatro, Tele 5, Sexta, TeleMadrid, Sexta 3 y *Paramount Channel*) extraída del contenido HTML de una página web mediante la utilización de la librería de Java JSoup. Se crea una tabla por cada canal de televisión. La aplicación Android permite al usuario actualizar dichas tablas con la programación del día de la consulta.

El usuario puede acceder a este sub-módulo interactuando con el sistema de varias formas:

- **Interacción oral:** Una vez dentro del módulo ***Asistente por voz*** y habiendo seleccionado acceder al módulo ***Asistente televisivo*** en modo oral, se dice al reconocedor de voz la palabra "Actualizar " y el usuario es dirigido al sub-módulo ***Actualización de la programación televisiva*** en modo oral.

- **Interacción multimodal:** Una vez dentro del menú de la Figura 4.19 después de haber accedido al módulo ***Asistente televisivo*** en modo multimodal, el usuario debe pulsar el botón de actualización o pulsar el botón  y decir la palabra "Actualizar" para acceder al sub-módulo ***Actualización de la programación televisiva*** en modo multimodal.

4.6.1.2. Arquitectura y flujo de datos

La Figura 4.20 muestra la arquitectura del sub-módulo ***Actualización de la programación televisiva***. Los pasos correspondientes al flujo de datos en la interacción multimodal se muestran en azul y los correspondientes a la interacción oral se muestran en verde.

Figura 4.20: Arquitectura y flujo de datos del sub-módulo Actualización de la programación televisiva

Interacción Multimodal

El proceso de actualización de la programación televisiva en modo multimodal se resume en los siguientes pasos:

1. La aplicación Android pide al usuario mediante la síntesis de texto a voz que indique el canal sobre el que desea solicitar la programación recomendada o bien que solicite la actualización de la base de datos.

2. El usuario indica a través de la interfaz o a través del reconocimiento de voz que desea actualizar la base de datos con la programación del día actual.

3. La aplicación Android (cliente) envía una petición mediante HTTP al servidor del sitio web Teletexto.com, Sexta 3 o Paramount Channel solicitando información del canal correspondiente.

4. El servidor del sitio web *teletexto.com*, Sexta 3 o *Paramount Channel* responde al cliente con el contenido HTML de la página web solicitada.

5. La aplicación extrae la información de la programación diaria, obtenida del contenido HTML del la página web, mediante la utilización de la librería JSoup.

6. La aplicación almacena la información en cuanto a la programación diaria en las tablas de programación televisiva de la base de datos SQLite.

Interacción Oral

El proceso de actualización de la programación televisiva en modo oral se resume en los siguientes pasos:

1. La aplicación Android pide al usuario mediante la síntesis de texto a voz que indique el canal sobre el que desea solicitar la programación recomendada o bien que solicite la actualización de la base de datos.

2. El usuario indica mediante el reconocimiento de voz que desea actualizar la base de datos con la programación del día actual.

3. La aplicación Android (cliente) envía una petición mediante HTTP al servidor del sitio web Teletexto.com, Sexta 3 o Paramount Channel solicitando información del canal correspondiente.

4. El servidor del sitio web *teletexto.com*, Sexta 3 o *Paramount Channel* responde al cliente con el contenido HTML de la página web solicitada.

5. La aplicación extrae la información de la programación diaria, obtenida del contenido HTML del la página web, mediante la utilización de la librería JSoup.

6. La aplicación almacena la información en cuanto a la programación diaria en las tablas de programación televisiva de la base de datos SQLite.

Para la implementación del presente sub-módulo se ha aplicado la metodología de inserción y manipulación de datos en una base de datos SQLite en Android vista en la Sección 3.2.2.2, la metodología para la extracción de contenido de las páginas web a través de la librería JSoup vista en la Sección 3.3.2, y la metodología para la la implementación de reconocimiento de voz y síntesis de texto a voz en aplicaciones para dispositivos móviles Android vista en la Sección 3.3.4.

4.6.1.3. Escenarios de uso

La Figura 4.21 muestra cómo se desarrolla el diálogo entre el usuario y la aplicación en el ***modo oral*** cuando se quiere actualizar la base de datos con la programación del día actual.

S: ¿Qué Servicio desea solicitar? Para escuchar las recomendaciones televisivas correspondientes a los gustos registrados diga: Asistente Televisivo. Para escuchar las recomendaciones de películas correspondientes a los gustos registrados diga: Recomendación de películas. Para buscar una película diga: Buscar película. Para solicitar la cartelera de su cine preferido diga: Cartelera.
U: Asistente televisivo.
(El usuario es dirigido al módulo ***Asistente televisivo***)
S: Seleccione el canal sobre el que desea escuchar las recomendaciones televisivas. Tenga en cuenta que deberá actualizar la base de datos sino se ha actualizado previamente durante el día de hoy. Para actualizar la base de datos diga: actualizar. Si quiere escuchar la programación para Televisión española diga: Televisión española. Si quiere escuchar la programación para La 2 diga: La 2. Si quiere escuchar la programación para Antena 3 diga: Antena 3. Si quiere escuchar la programación para La cuatro diga: La cuatro. Si quiere escuchar la programación para Telecinco diga: Telecinco. Si quiere escuchar la programación para la sexta diga: la sexta. Si quiere escuchar la programación para Telemadrid diga: Telemadrid. Si quiere escuchar la programación para La sexta 3 diga: La sexta 3. Si quiere escuchar la programación para el canal *Paramount channel* diga: *Paramount channel*.
U: Actualizar. (El usuario es dirigido al sub-módulo ***Actualización de la programación televisiva*** y se actualiza la base de datos con la programación del día actual)

Figura 4.21: Diálogo establecido entre el usuario y la aplicación en el sub-módulo Actualización de la programación televisiva en modo oral

4.6.2. Sub-módulo Recomendación de la programación televisiva

4.6.2.1. Funcionalidad

El sub-módulo ***Recomendación de programación televisiva*** se encarga de consultar la base de datos SQLite con el fin de obtener información de la programación diaria de las tablas de programación televisiva para cada canal e información sobre los gustos televisivos del usuario de la tabla de gustos televisivos. Posteriormente procesa la información obtenida de las tablas para devolver al usuario la programación diaria asociada a los gustos registrados.

El usuario puede acceder a este sub-módulo interactuando con el sistema de varias formas:

- **Interacción oral:** Una vez dentro del módulo ***Asistente por voz*** y habiendo seleccionado acceder al módulo ***Asistente televisivo*** en modo oral, se dice al reconocedor de voz el nombre del canal sobre el que se desea solicitar información y el usuario es dirigido al sub-módulo ***Recomendación de programación televisiva*** en modo oral.

- **Interacción multimodal:** Una vez dentro del menú de la Figura 4.19 después de haber accedido al módulo ***Asistente televisivo*** en modo multimodal, el usuario debe pulsar el botón asociado al canal sobre el que se desea solicitar información o pulsar el botón  y decir el canal y el usuario es dirigido al sub-módulo ***Recomendación de programación televisiva*** en modo multimodal.

El sub-módulo ***Recomendación de programación televisiva*** en **modo multimodal** se realiza únicamente en el lado del cliente por lo que no es necesario tener acceso a Internet si no se utiliza el reconocimiento de voz como entrada al sistema. Además, el usuario puede consultar información más detallada de cada película, documental y serie de TV recomendada por el asistente.

Por otro lado, el sub-módulo ***Recomendación de programación televisiva*** en **modo oral** necesita acceso a Internet ya que requiere utilizar el reconocimiento de voz en todo momento y además no permite al usuario solicitar información más detallada sobre la película recomendada.

Para poder utilizar el servicio ofrecido por el sub-módulo ***Recomendación de programación televisiva*** es necesario haber registrado los gustos de televisión en el cuestionario de registro.

4.6.2.2. Arquitectura y flujo de datos

La Figura 4.22 muestra la arquitectura y proceso del sub-módulo ***Recomendación de programación televisiva***. Dicho proceso se inicia cuando el usuario selecciona un canal de televisión y se resume en los siguientes pasos. Los pasos correspondientes al flujo de datos en la interacción multimodal se muestran en azul y los correspondientes a la interacción oral se muestran en verde.

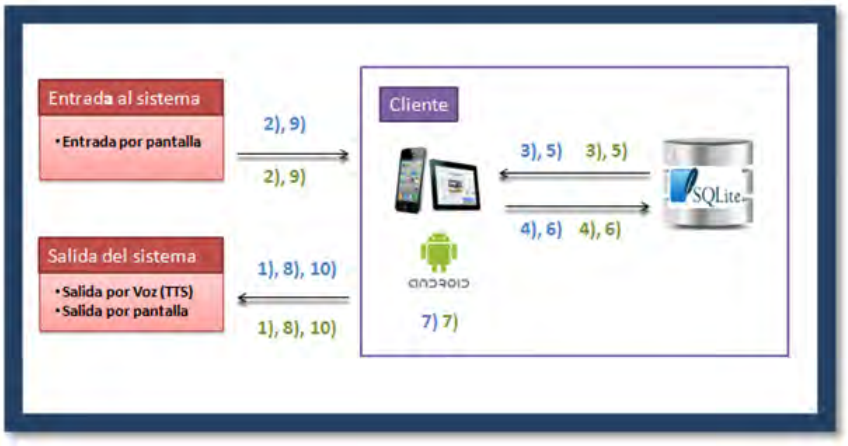


Figura 4.22: Arquitectura y flujo de datos del sub-módulo Recomendación de programación televisiva

Interacción Multimodal

1. La aplicación Android pide al usuario mediante la síntesis de texto a voz que indique el canal sobre el que desea solicitar la programación recomendada o bien que solicite la actualización de la base de datos.
2. El usuario comunica a la aplicación a través de la interfaz gráfica o a través del reconocimiento de voz el canal sobre el que desea solicitar la programación recomendada.
3. La aplicación Android solicita a la base de datos SQLite los gustos sobre la programación televisiva registrados por el usuario y almacenados en la tabla de gustos televisivos.
4. La base de datos SQLite devuelve los datos solicitados a la aplicación Android.
5. La aplicación Android solicita a la base de datos SQLite la información de la programación televisiva almacenada en la tabla de programación televisiva para el canal seleccionado por el usuario.
6. La base de datos SQLite devuelve los datos solicitados a la aplicación Android.
7. La aplicación Android procesa la información recuperada de ambas tablas y obtiene la programación televisiva recomendada según los gustos registrados por el usuario y para el canal seleccionado.
8. La aplicación Android muestra por pantalla la categoría de resultados que ha encontrado de acuerdo con los gustos registrados. Además, la aplicación utiliza la síntesis de texto a voz para informar al usuario de dichos resultados e indicarle que seleccione la categoría sobre la que desea solicitar la parrilla televisiva.
9. El usuario comunica a la aplicación a través de la interfaz gráfica o a través del reconocimiento de voz la categoría sobre la que desea solicitar la programación televisiva.
10. La aplicación muestra por pantalla y reproduce mediante la síntesis de texto a voz la programación televisiva de acuerdo con la categoría seleccionada.

Finalmente, el asistente televisivo permite al usuario solicitar información más detallada sobre las películas y series recomendadas. El proceso y arquitectura son los mismos que utiliza el servicio de búsqueda de películas por lo que es necesario disponer de acceso a Internet para poder solicitar dicha información.

Interacción Oral

1. La aplicación Android pide al usuario mediante la síntesis de texto a voz que indique el canal sobre el que desea solicitar la programación recomendada o bien que solicite la actualización de la base de datos.
2. El usuario comunica a la aplicación mediante el reconocimiento de voz el canal sobre el que desea solicitar la programación recomendada.
3. La aplicación Android solicita a la base de datos SQLite los gustos sobre la programación televisiva registrados por el usuario y almacenados en la tabla de gustos televisivos.
4. La base de datos SQLite devuelve los datos solicitados a la aplicación Android.
5. La aplicación Android solicita a la base de datos SQLite la información de la programación televisiva almacenada en la tabla de programación televisiva para el canal seleccionado por el usuario.

6. La base de datos SQLite devuelve los datos solicitados a la aplicación Android.
7. La aplicación Android procesa la información recuperada de ambas tablas y obtiene la programación televisiva recomendada según los gustos registrados por el usuario y para el canal seleccionado.
8. La aplicación Android utiliza la síntesis de texto a voz para informar al usuario de dichos resultados e indicarle que seleccione la categoría sobre la que desea solicitar la parrilla televisiva.
9. El usuario comunica a la aplicación a través del reconocimiento de voz la categoría sobre la que desea solicitar la programación televisiva.
10. La aplicación reproduce mediante la síntesis de texto a voz la programación televisiva de acuerdo con la categoría seleccionada.

Para la implementación del presente sub-módulo se ha aplicado la metodología de acceso a una base de datos SQLite en Android vista en la Sección 3.2.2.2 y la metodología para la la implementación de reconocimiento de voz y síntesis de texto a voz en aplicaciones para dispositivos móviles Android vista en la Sección 3.3.4.

4.6.2.3. Escenarios de uso

La Figura 4.23 muestra, a través de capturas de pantalla sacadas de la aplicación Android, un ejemplo de utilización en **modo multimodal** del servicio del sub-módulo ***Recomendación de programación televisiva***. Tal y como se observa, los resultados mostrados se corresponden con las categorías de cine y documentales, al ser estos los gustos televisivos registrados por el usuario tal y como se observa en la Figura 4.11 y Figura 4.12.

A continuación, se describe cada una de las capturas de pantalla que se van mostrando de manera secuencial al usuario durante la utilización del servicio.

1. Menú que permite al usuario seleccionar un canal. El usuario puede indicar a la aplicación el canal sobre el que desea solicitar información mediante el reconocimiento de voz pulsando el botón n o pulsando la entrada correspondiente en el menú. El usuario selecciona “La Sexta”. Además, si se pulsa el botón , se inicia la síntesis de texto a voz que emite las instrucciones. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz.
2. Muestra la categoría de resultados encontrados. Además, si se pulsa el botón , se inicia la síntesis de texto a voz que emite la categoría de resultados encontrados y le pide al usuario que seleccione una de las categorías mediante el reconocimiento de voz o pulsando la entrada correspondiente. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz. Si el usuario pulsa el botón puede decir al reconocedor de voz la categoría sobre la que desea escuchar las recomendaciones.
3. Muestra la programación de acuerdo con la categoría seleccionada.
   a) Muestra la programación relacionada con los documentales.
   b) Muestra la programación relacionada con el cine. Si se pulsa sobre alguna de las películas el usuario es dirigido al módulo de “Búsqueda de películas”. En el caso del ejemplo se ha seleccionado la película *American History X*.
4. Resultado de la película solicitada. Si se pulsa el botón se inicia la actividad de síntesis de texto a voz que informa al usuario sobre el título de la película solicitada y le pide que solicite la información que desea saber acerca de dicha película. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz. El usuario puede seleccionar la opción o bien mediante el reconocimiento de voz pulsando el botón , o bien mediante el teclado seleccionando una de las opciones que se ofrecen.

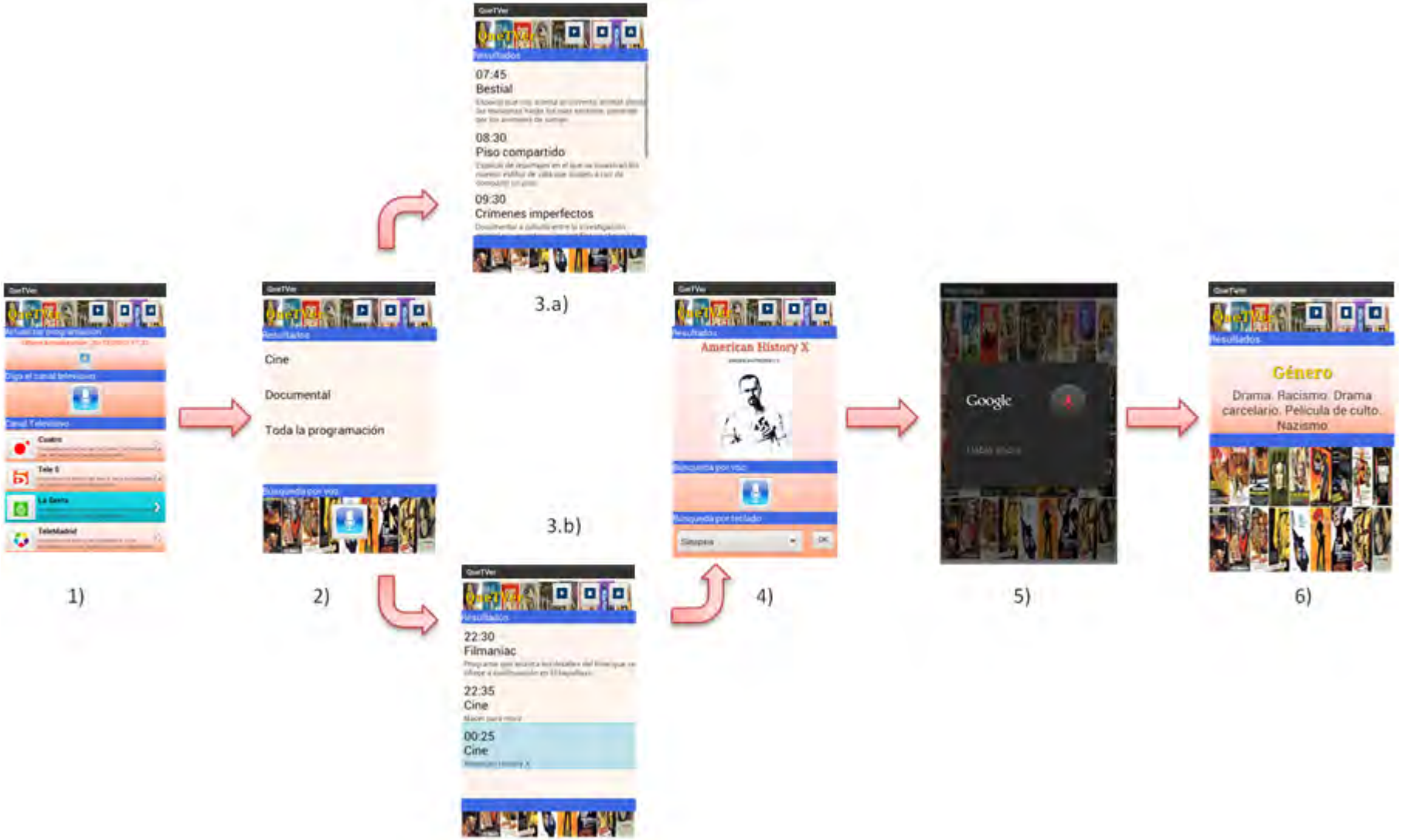


Figura 4.23: Ejemplo de utilización del sub-módulo Recomendación de programación televisiva en modo multimodal

5. El usuario indica a la aplicación la información que desea solicitar acerca de la película encontrada. Las opciones son: sinopsis, título original, país, duración, director, reparto, género, año, calificación, premios. En el caso del ejemplo ha decido utilizar el reconocedor de voz para solicitar el género de la película seleccionada.

6. Muestra el resultado en cuanto al género de la película *American History X*. Al pulsarse el boton ▶ se inicia la lectura del reparto. En cualquier momento se puede pulsar el botón ■ y se detiene la síntesis de texto a voz.

La Figura 4.24 muestra cómo se desarrolla el diálogo de la aplicación en el **modo oral** para el ejemplo de la Figura 4.23.

S: ¿Qué Servicio desea solicitar? Para escuchar las recomendaciones televisivas correspondientes a los gustos registrados diga: Asistente Televisivo. Para escuchar las recomendaciones de películas correspondientes a los gustos registrados diga: Recomendación de películas. Para buscar una película diga: Buscar película. Para solicitar la cartelera de su cine preferido diga: Cartelera.

U: Asistente Televisivo.

(El usuario es dirigido al módulo ***Asistente televisivo***)

S: Seleccione el canal sobre el que desea escuchar las recomendaciones televisivas. Tenga en cuenta que deberá actualizar la base de datos sino se ha actualizado previamente durante el día de hoy. Para actualizar la base de datos diga: actualizar. Si quiere escuchar la programación para Televisión española diga: Televisión española. Si quiere escuchar la programación para La 2 diga: La 2. Si quiere escuchar la programación para Antena 3 diga: Antena 3. Si quiere escuchar la programación para La cuatro diga: La cuatro. Si quiere escuchar la programación para Telecinco diga: Telecinco. Si quiere escuchar la programación para la sexta diga: la sexta. Si quiere escuchar la programación para Telemadrid diga: Telemadrid. Si quiere escuchar la programación para La sexta 3 diga: La sexta 3. Si quiere escuchar la programación para el canal *Paramount channel* diga: *Paramount channel*.

U: La Sexta.

S: Se han encontrado los siguientes resultados correspondientes a sus gustos: Cine y Documentales. Diga al reconocedor la opción sobre la que desea saber información, o bien, seleccione la opción de la lista.

U: Cine.

S: Como programación de cine se le recomienda: A las 22:30 Filmaniac, programa que analiza los detalles del filme que se ofrece a continuación en El taquillazo. A las 22:35 Cine, Nacer para morir. A las 00:25 cine, American History X.

Figura 4.24: Diálogo establecido entre el usuario y la aplicación en el sub-módulo Recomendación de programación televisiva en modo oral

4.7. Módulo Recomendación de películas

4.7.1. Funcionalidad

El módulo ***Recomendación de películas*** recomienda al usuario películas de acuerdo con los gustos registrados. Tal y como se ha explicado, una vez que el usuario registra sus gustos de cine a través del formulario inicial, se genera la tabla de recomendación de películas en la base de datos SQLite que contiene la lista de películas recomendadas al usuario. Cada vez que el usuario solicita el servicio de recomendación de películas, dicho servicio recomienda al usuario una película de la lista.

El usuario puede acceder a este módulo interactuando con el sistema de varias formas:

- **Interacción oral:** Mediante la opción del menú **Asistente por voz** del menú principal, que conduce al usuario al módulo ***Recomendación de películas*** si en su primera decisión el usuario dice al reconocedor la palabra "Recomendar película", "Recomendar películas" o "Recomendación de películas".

- **Interacción multimodal:** El módulo ***Recomendación de películas*** en modo multimodal, se accede desde el menú del modulo ***Inicio*** mostrado en la Figura 4.8. El usuario puede o bien pulsar la opción del menú **Recomendación de películas** o bien decir la palabra clave "Recomendación de películas" al reconocedor de voz pulsando el botón  . En todo momento, la aplicación permite al usuario seleccionar la modalidad con la que desea interactuar con el dispositivo de acuerdo a sus preferencias y a las condiciones del ambiente de uso. El usuario puede interactuar con el sistema mediante el modo táctil, el modo oral, o combinar ambos modos.

El módulo ***Recomendación de películas*** en **modo multimodal** funciona en modo *offline* y permite al usuario solicitar información más detallada acerca de la película recomendada. No obstante, el módulo ***Recomendación de películas*** en **modo oral** necesita acceso a Internet ya que requiere utilizar el reconocimiento de voz para activarlo y además no permite al usuario solicitar información más detallada sobre la película recomendada.

Para poder utilizar el servicio ofrecido por el módulo ***Recomendación de películas*** es necesario haber registrado los gustos de cine en el cuestionario de registro.

4.7.2. Arquitectura y flujo de datos

La arquitectura del módulo ***Recomendación de películas*** se muestra en la Figura 4.25. Los pasos correspondientes al flujo de datos en la interacción multimodal se muestran en azul y los correspondientes a la interacción oral se muestran en verde.

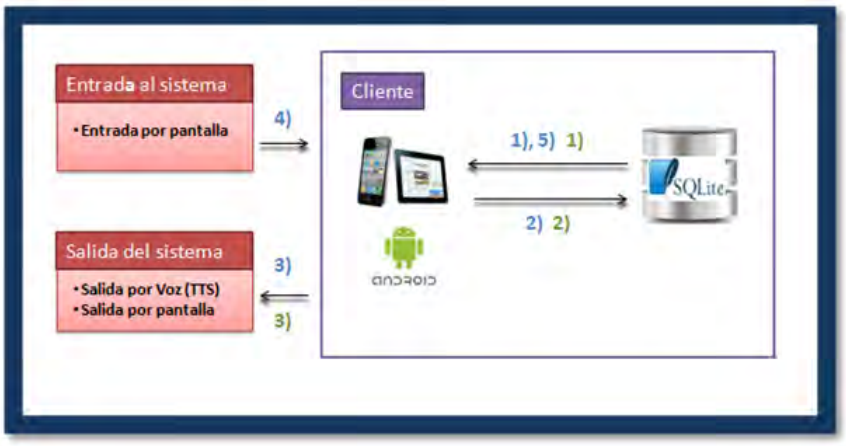


Figura 4.25: Arquitectura y flujo de datos en el módulo Recomendación de películas

Interacción Multimodal

El proceso llevado a cabo cuando se solicita acceder a este módulo mediante interacción multimodal se resume en los pasos listados a continuación:

1. La aplicación Android (cliente) accede a la base de datos SQLite para consultar la primera película recomendada según la tabla de películas recomendadas.
2. La base de datos SQLite devuelve el título y año de la película recomendada al usuario según sus gustos registrados.
3. La aplicación Android muestra por pantalla el título y año de la película recomendada. Además reproduce el título y año de la película mediante la síntesis de texto a voz.
4. El usuario solicita a la aplicación mediante la interfaz gráfica que le recomiende otra película.
5. La aplicación Android accede a la base de datos SQLite para consultar la siguiente película recomendada según la tabla de películas recomendadas. Se va al paso 2 y se repite el proceso hasta que el usuario no desee solicitar más recomendaciones.

El módulo ***Recomendación de películas*** en modo multimodal permite al usuario solicitar información más detallada sobre la película recomendada. El proceso y la arquitectura son los mismos que utiliza el módulo ***Búsqueda de películas*** por lo que es necesario disponer de acceso a Internet para poder solicitar dicha información.

Interacción Oral

El proceso llevado a cabo cuando se solicita acceder a este módulo mediante interacción oral se resume en los pasos listados a continuación:

1. La aplicación Android (cliente) accede a la base de datos SQLite para consultar la lista de películas recomendadas según la tabla de películas recomendadas.
2. La base de datos SQLite devuelve el título y año de las películas recomendadas al usuario según sus gustos registrados.
3. La aplicación Android reproduce el título y año de las películas mediante la síntesis de texto a voz.

El módulo ***Recomendación de películas*** en modo oral no permite al usuario solicitar información más detallada sobre la película recomendada.

Para la implementación del presente sub-módulo se ha aplicado la metodología de acceso a una base de datos SQLite en Android vista en la Sección 3.2.2.2 y la metodología para la la implementación de reconocimiento de voz y síntesis de texto a voz en aplicaciones para dispositivos móviles Android vista en la Sección 3.3.4.

4.7.3. Escenarios de uso

La Figura 4.26 muestra, a través de una serie de capturas de pantalla sacadas de la aplicación, un ejemplo de utilización del módulo ***Recomendación de películas*** en **modo multimodal**. Tal y como se observa, las películas recomendadas se corresponden con los gustos registrados en la Figura 4.11 y Figura 4.12. Se han encontrado resultados de cine bélico, con país de origen EEUU, con año de origen 1950 y año destino 2000, y se han excluido las series y documentales de los resultados.

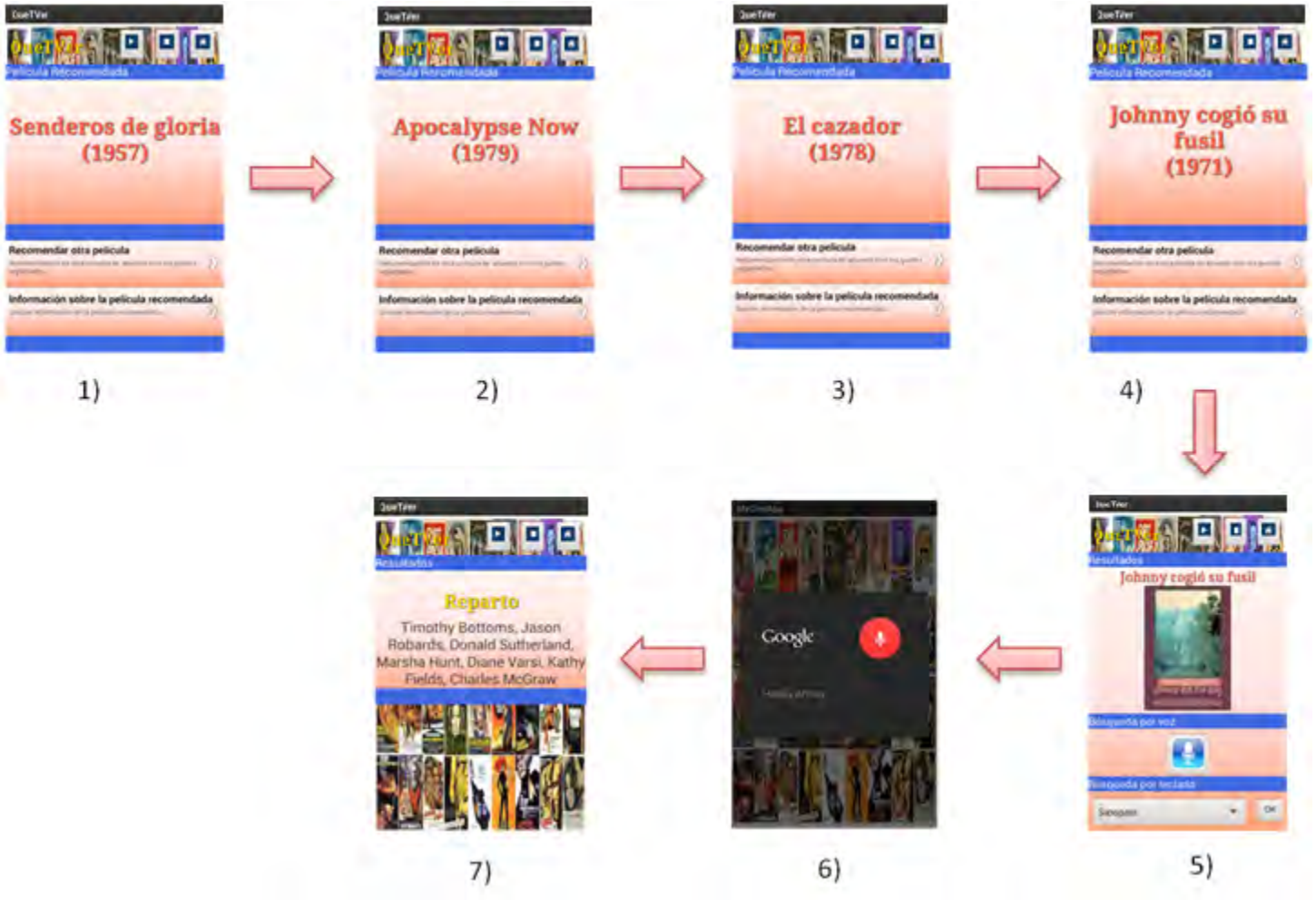


Figura 4.26: Ejemplo de utilización del módulo Recomendación de películas en modo multimodal

A continuación, se describe cada una de las capturas de pantalla que se van mostrando de manera secuencial al usuario durante la utilización del servicio.

1. La aplicación muestra por pantalla el título y año de la primera película recomendada y lo reproduce mediante la síntesis de texto a voz. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz y el botón para reanudarla. El usuario pulsa la opción "Recomendar otra película".

2. La aplicación muestra por pantalla el título y año de la siguiente película recomendada y lo reproduce mediante la síntesis de texto a voz. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz y el botón para reanudarla. El usuario pulsa la opción "Recomendar otra película".

3. La aplicación muestra por pantalla el título y año de la siguiente película recomendada y lo reproduce mediante la síntesis de texto a voz. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz y el botón para reanudarla. El usuario pulsa la opción "Recomendar otra película".

4. La aplicación muestra por pantalla el título y año de la siguiente película recomendada y lo reproduce mediante la síntesis de texto a voz. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz y el botón para reanudarla. El usuario pulsa la opción "Información acerca de la película recomendada".

5. Resultado de la película solicitada. Si se pulsa el botón se inicia la actividad de síntesis de texto a voz que informa al usuario sobre el título de la película solicitada y le pide que solicite la información que desea saber acerca de dicha película. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz. El usuario puede seleccionar la opción o bien mediante el reconocimiento de voz pulsando el botón , o bien mediante el teclado seleccionando una de las opciones que se ofrecen.

6. El usuario indica a la aplicación la información que desea solicitar acerca de la película encontrada. Las opciones son: sinopsis, título original, país, duración, director, reparto, género, año, calificación, premios. En el caso del ejemplo ha decido utilizar el reconocedor de voz para solicitar la sinopsis de la película seleccionada.

7. El reparto de la película *Johnny cogió su fusil*. Al pulsarse el botón se inicia la lectura del reparto. Al detectar que se trata de una película americana el reparto se dirá con acento inglés. En cualquier momento se puede pulsar el botón y se detiene la síntesis de texto a voz.

La Figura 4.27 muestra cómo se desarrolla el diálogo entre el usuario y la aplicación en el ***modo oral***.

S: ¿Qué Servicio desea solicitar? Para escuchar las recomendaciones televisivas correspondientes a los gustos registrados diga: Asistente Televisivo. Para escuchar las recomendaciones de películas correspondientes a los gustos registrados diga: Recomendación de películas. Para buscar una película diga: Buscar película. Para solicitar la cartelera de su cine preferido diga: Cartelera.
U: Recomendación de películas.
(El usuario es dirigido al servicio ***Recomendación de películas***)
S: Senderos de gloria: año 1957, Apocalypse Now: año 1979, El cazador: año 1978, Johnny cogió su fusil: año 1971, ...

Figura 4.27: Diálogo establecido entre el usuario y la aplicación en el módulo Recomendación de películas en el modo oral

4.8. Módulo Cartelera

4.8.1. Funcionalidad

El módulo ***Cartelera*** permite al usuario buscar salas de cine y obtener información de la cartelera asociada a dicho cine. La búsqueda se realiza indicando el nombre de la provincia y ciudad sobre la que se desea obtener información de las salas de cine. Dicha información se obtiene dinámicamente del sitio web *hoyCinema.abs.es* por lo que es necesario tener acceso a Internet para poder utilizar el servicio.

El usuario puede acceder a este módulo interactuando con el sistema de varias formas:

- **Interacción oral:** Mediante la opción del menú **Asistente por voz** del menú principal, que conduce al usuario al módulo ***Cartelera*** si en su primera decisión el usuario dice al reconocedor la palabra "Cartelera".
- **Interacción multimodal:** El módulo ***Cartelera*** en modo multimodal, se accede desde el menú del modulo ***Inicio*** mostrado en la Figura 4.8. El usuario puede o bien pulsar la opción del menú **Cartelera** o bien decir la palabra clave "Cartelera" al reconocedor de voz pulsando el botón . En todo momento, la aplicación permite al usuario seleccionar la modalidad con la que desea interactuar con el dispositivo de acuerdo a sus preferencias y a las condiciones del ambiente de uso. El usuario puede interactuar con el sistema mediante el modo táctil, el modo oral, o combinar ambos modos.

El módulo ***Cartelera*** en **modo multimodal** permite al usuario solicitar información más detallada acerca de la película recomendada. No obstante, el módulo ***Recomendación de películas*** en **modo oral** no permite al usuario solicitar información más detallada sobre la película recomendada.

4.8.2. Arquitectura y flujo de datos

La Figura 4.28 muestra la arquitectura y el correspondiente flujo de datos del módulo ***Cartelera***. Los pasos correspondientes al flujo de datos en la interacción multimodal se muestran en azul y los correspondientes a la interacción oral se muestran en verde.

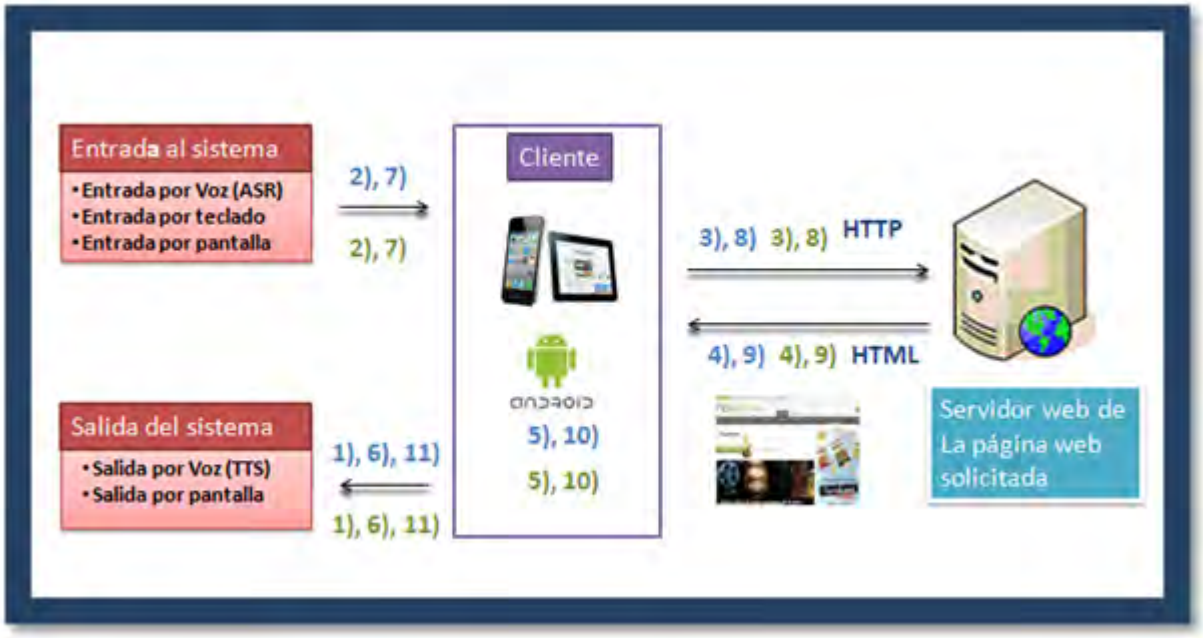


Figura 4.28: Flujo de datos del módulo Cartelera

Interacción Multimodal

El proceso llevado a cabo cuando se solicita acceder a este módulo mediante interacción multimodal se resume en los pasos listados a continuación:

1. La aplicación Android pide al usuario mediante la síntesis de texto a voz que indique la provincia del cine del cual desea solicitar su cartelera.

2. El usuario comunica a la aplicación a través de la interfaz gráfica o a través del reconocimiento de voz la provincia del cine del cual desea solicitar su cartelera.

3. La aplicación Android (cliente) envía una petición mediante HTTP al servidor del sitio web de *hoyCinema.abs.es* y se conecta a la página web que muestra las salas de cine situadas en la provincia que ha seleccionado el usuario.

4. El servidor del sitio web de *hoyCinema.abs.es* responde al cliente con el contenido HTML de la página web que muestra las salas de cine.

5. La aplicación extrae la información de las salas encontradas en la provincia y ciudad solicitada por el usuario, obtenida del contenido HTML de su página web mediante la utilización de la librería JSoup. La aplicación Android muestra por pantalla las salas de cine encontradas.

6. La aplicación Android solicita al usuario mediante la síntesis de texto a voz que seleccione la sala de la que desea solicitar la cartelera.

7. El usuario selecciona mediante la interfaz gráfica de la aplicación el cine sobre el que desea solicitar la cartelera.

8. La aplicación Android envía una petición mediante HTTP al servidor del sitio web de *hoyCinema.abs.es* y se conecta a la página web que muestra la cartelera de cine de la sala seleccionada por el usuario.

9. El servidor del sitio web de *hoyCinema.abs.es* responde al cliente con el contenido HTML de la página web que muestra la cartelera de dicho cine.

10. La aplicación extrae la información de la cartelera, obtenida del contenido HTML de la página web mediante la utilización de la librería JSoup. La aplicación Android muestra por pantalla la cartelera. Para mostrar las imágenes correspondientes a la cartelera, se utiliza el proceso descrito en la Sección 3.3.3 para la carga asíncrona de imágenes remotas y su almacenamiento en la caché del dispositivo.

11. La aplicación muestra por pantalla o reproduce mediante la síntesis de texto a voz la cartelera.

El usuario puede solicitar información de cualquier película de la cartelera. Para ello, debe seleccionar una de sus peliculas de la cartelera mostrada por pantalla. El usuario será dirigido al módulo ***Búsqueda de películas*** cuya entrada será la película seleccionada.

Interacción Oral

El proceso llevado a cabo cuando se solicita acceder a este módulo mediante interacción oral se resume en los pasos listados a continuación:

1. La aplicación Android pide al usuario mediante la síntesis de texto a voz que indique la provincia del cine del cual desea solicitar su cartelera.

2. El usuario comunica a la aplicación a través del reconocimiento de voz la provincia del cine del cual desea solicitar su cartelera.

3. La aplicación Android (cliente) envía una petición mediante HTTP al servidor del sitio web de *hoyCinema.abs.es* y se conecta a la página web que muestra las salas de cine situadas en la provincia que ha seleccionado el usuario.

4. El servidor del sitio web de *hoyCinema.abs.es* responde al cliente con el contenido HTML de la página web que muestra las salas de cine.

5. La aplicación extrae la información de las salas encontradas en la provincia y ciudad solicitada por el usuario, obtenida del contenido HTML de su página web mediante la utilización de la librería JSoup.

6. La aplicación Android solicita al usuario mediante la síntesis de texto que seleccione la sala de la que desea solicitar la cartelera. La aplicación asocia un número a cada elemento de la lista y pide al usuario, mediante la síntesis de texto a voz, que le diga al reconocedor de voz el número asociado a la sala de cine sobre la que desea solicitar información.

7. El usuario selecciona mediante la síntesis de texto a voz el número asociado a la sala de cine sobre la que desea solicitar información.

8. La aplicación Android envía una petición mediante HTTP al servidor del sitio web de *hoyCinema.abs.es* y se conecta a la página web que muestra la cartelera de cine de la sala seleccionada por el usuario.

9. El servidor del sitio web de *hoyCinema.abs.es* responde al cliente con el contenido HTML de la página web que muestra la cartelera de dicho cine.

10. La aplicación extrae la información de la cartelera, obtenida del contenido HTML de la página web mediante la utilización de la librería JSoup.

11. La aplicación reproduce mediante la síntesis de texto a voz la cartelera.

Para la implementación de este módulo se han aplicado la extracción de contenido de las páginas web a través de la librería JSoup descritas en la Sección 3.3.2, la carga asíncrona de imágenes remotas descrita en la Sección 3.3.3 y la implementación de reconocimiento de voz y síntesis de texto a voz en aplicaciones para dispositivos móviles Android descrita en la Sección 3.3.4.

4.8.3. Escenarios de uso

La Figura 4.29 muestra, a través de una serie de capturas de pantalla sacadas de la aplicación, un ejemplo de utilización del módulo ***Cartelera*** en modo multimodal.

A continuación, se describe cada una de las capturas de pantalla que se van mostrando de manera secuencial al usuario durante la utilización del servicio.

1. Menú que permite al usuario especificar la provincia en la que se encuentra el cine del que se solicita la cartelera. El usuario puede especificar la provincia mediante su selección en la lista que se muestra por pantalla o mediante la utilización del servicio de reconocimiento de voz pulsando el botón  . Además, si se pulsa el botón  , se inicia la síntesis de texto a voz que emite las instrucciones. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz.

2. El usuario especifica la provincia en la que se encuentra el cine del que se solicita la cartelera o bien mediante la interfaz gráfica o mediante el reconocimiento de voz. En el caso del ejemplo la provincia especificada es “Cuenca”.

    a) El usuario dice la provincia en la que se encuentra el cine del que se solicita la cartelera al reconocedor de voz.

b) El usuario selecciona de la lista de provincias la provincia en la que se encuentra el cine del que se solicita la cartelera.

3. Lista de cines encontrados para la localidad especificada previamente. El usuario selecciona la sala de cine de la lista sobre la que desea solicitar la cartelera. En el caso del ejemplo selecciona la sala "Ábaco Cuenca". Además, si se pulsa el botón , se inicia la síntesis de texto a voz que emite las instrucciones y los resultados de la lista. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz.

4. Muestra una lista por pantalla con la cartelera de la sala de cine especificada previamente. Si se pulsa el botón se inicia la actividad de síntesis de texto a voz que informa al usuario sobre el título de las películas en cartelera para la sala de cine especificada previamente. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz. El usuario puede solicitar información de cualquier película de la lista. Al seleccionar una de las películas de la lista, la aplicación dirige al usuario al módulo ***Búsqueda de películas*** y busca información acerca de la película seleccionada. En el caso del ejemplo se selecciona la película "3 días para matar".

5. Resultado de la película solicitada. Si se pulsa el botón se inicia la actividad de síntesis de texto a voz que informa al usuario sobre el título de la película solicitada y le pide que solicite la información que desea saber acerca de dicha película. En cualquier momento se puede pulsar el botón para detener la síntesis de texto a voz. El usuario puede seleccionar la opción o bien mediante el reconocimiento de voz pulsando el botón , o bien mediante el teclado seleccionando una de las opciones que se ofrecen.

6. El usuario indica a la aplicación la información que desea solicitar acerca de la película encontrada. Las opciones son: sinopsis, título original, país, duración, director, reparto, género, año, calificación, premios.

   a) El usuario dice al reconocedor de voz la información que desea solicitar acerca de la película encontrada.

   b) El usuario selecciona de la lista la información que desea solicitar acerca de la película encontrada.

7. Información solicitada por el usuario de la película encontrada. En el caso del ejemplo, el usuario a solicitado la duración de la película "3 días para matar". Al pulsarse el botón se inicia la lectura de la sinopsis. En cualquier momento se puede pulsar el botón y se detiene la síntesis de texto a voz.

La Figura 4.30 muestra cómo se desarrolla el diálogo de la aplicación para el mismo ejemplo que el de la Figura 4.29.

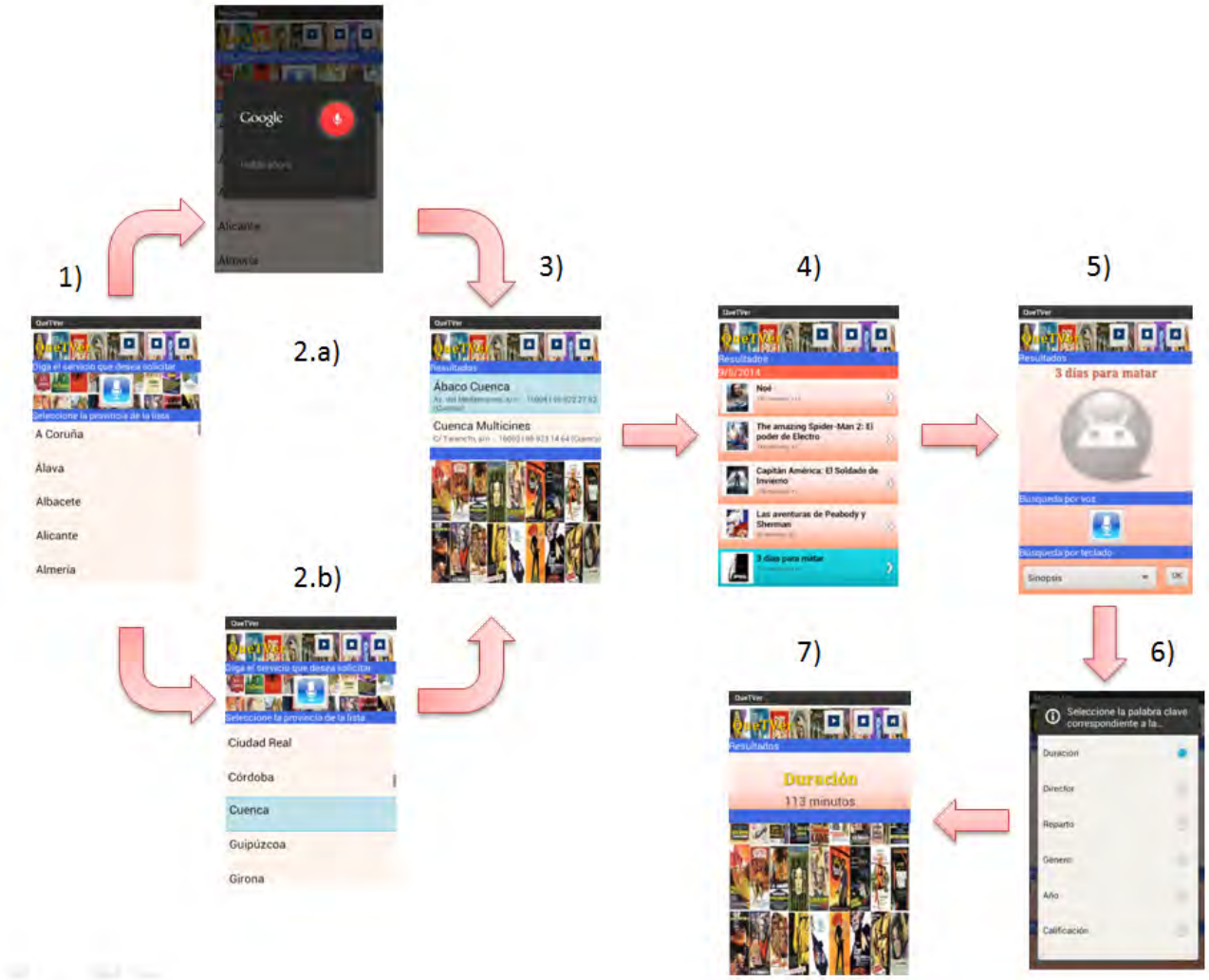


Figura 4.29: Ejemplo del módulo Cartelera en modo multimodal

S: ¿Qué Servicio desea solicitar? Para escuchar las recomendaciones televisivas correspondientes a los gustos registrados diga: Asistente Televisivo. Para escuchar las recomendaciones de películas correspondientes a los gustos registrados diga: Recomendación de películas. Para buscar una película diga: Buscar película. Para solicitar la cartelera de su cine preferido diga: Cartelera.

U: Cartelera.

(El usuario es dirigido al servicio ***Cartelera***)

S: Diga la provincia en la que se encuentra la sala de cine sobre la que desea solicitar la cartelera.

U: Cuenca.

S: Se han encontrado dos salas de cine para la provincia seleccionada. Para seleccionar la cartelera sobre el cine Ábaco Cuenca diga: 1. Para seleccionar la cartelera sobre el cine Cuenca Multicines diga: 2.

U: 1.

S: Se han encontrado las siguientes películas en cartelera: Noé, The amazing Spider-Man 2: El poder de Electro, Capitán América: El Soldado de Invierno, Las aventuras de Peabody y Sherman, 3 días para matar, Tio 2, Malditos vecinos, Divergente, Pompeya.

Figura 4.30: Diálogo establecido entre el usuario y la aplicación en el módulo Cartelera en modo oral

Capítulo 5

EVALUACIÓN DEL SISTEMA

En el presente capítulo se describe la metodología utilizada en la evaluación de la aplicación multimodal desarrollada para el Proyecto Final de Carrera, basada en un cuestionario que recoge las valoraciones subjetivas de sus usuarios desde una página web. Los resultados se almacenan en una base de datos y posteriormente se analizan para obtener conclusiones.

5.1. Metodología de evaluación

La metodología utilizada para evaluar la aplicación multimodal desarrollada en el presente Proyecto Final de Carrera se ha basado en un artículo que trata de la evaluación fiable de sistemas de diálogo multimodales. Según dicho artículo (Metze et al., 2009), todo desarrollo y despliegue de un sistema de diálogo multimodal requiere probar el rendimiento del sistema en sus distintas modalidades. Para ello, se realiza un análisis de la usabilidad del sistema a través de un cuestionario, cuyas preguntas están orientadas a evaluar el sistema multimodal además de la interacción mediante el modo táctil o mediante la voz.

La Figura 5.1 muestra el cuestionario elaborado para evaluar el sistema en sus diferentes modalidades. El cuestionario consta de 23 preguntas. Las preguntas 3-9 corresponden a la evaluación de la interacción mediante el modo táctil, las preguntas 10-16 corresponden a la evaluación de la interacción mediante el modo oral, y las preguntas 17-23 corresponden a la evaluación de la interacción mediante las diferentes modalidades de entrada y/salida. Cada pregunta tiene 5 posibles respuestas de las que sólo se puede elegir una de ellas.

Tal y como se observa, los aspectos que se han querido analizar para las distintas modalidades del sistema son: el grado en el cual el usuario valora que es entendido por el sistema y entiende los mensajes del mismo, la velocidad percibida de la de interacción, el nivel de dificultad del sistema, la presencia de errores, la seguridad de lo que se debe hacer en cada momento, y el nivel de satisfacción con el sistema global. Además, información adicional de los usuarios sobre su grado de conocimiento y experiencia previa con los interfaces multimodales y los interfaces orales sirven para hacerse una idea del perfil de estos usuarios.

1. Puntúe en una escala de 1 a 5 su experiencia previa usando Interfaces de voz (1= “Bajo”, 5= “Alto”).

- 1
- 2
- 3
- 4
- 5

2. Puntúe en una escala de 1 a 5 su experiencia previa usando Interfaces multimodales (1= “Bajo”, 5= “Alto”).

- 1
- 2
- 3
- 4
- 5

Evaluación del sistema mediante los interfaces tradicionales: Acceda a las opciones del menú “Asistente televisivo”, “Recomendación de películas”, “Buscar películas” y “Cartelera” utilizando únicamente la interfaz gráfica y el teclado.

3. ¿Qué tal ha entendido el sistema sus peticiones?

- Muy mal
- Mal
- Regular
- Bien
- Muy bien

4. ¿Cómo ha entendido los mensajes generados por el sistema?

- Muy mal
- Mal
- Regular
- Bien
- Muy bien

5. En su opinión la interacción fue...

- Muy lenta
- Lenta
- Adecuada
- Rápida
- Muy rápida

6. Establezca el nivel de dificultad del sistema en modo táctil para usted.

- Muy difícil
- Difícil
- Normal
- Fácil
- Muy fácil

7. ¿Ha percibido errores durante su interacción con el sistema en modo táctil?

- Sí, he percibido muchos errores lo que ha imposibilitado la interacción con el sistema
- Sí, he percibido bastantes errores lo que ha causado gran dificultad en la interacción con el sistema
- Sí, he percibido algunos errores lo que ha causado cierta dificultad en la interacción con el sistema
- Sí, he percibido algunos errores aunque la interacción con el sistema no se ha visto afectada en absoluto
- No he percibido errores

8. ¿Le ha sido fácil decidir que hacer en cada momento para obtener la información que usted solicitaba?

- No, ha sido imposible
- No, me ha supuesto gran dificultad
- Sí, pero con ciertas dificultades
- Sí, ha sido fácil
- Sí, ha sido muy fácil

9. En términos generales, ¿está usted satisfecho con el sistema en modo táctil?

- No, nada
- Poco satisfecho
- Satisfecho
- Bastante satisfecho
- Muy satisfecho

Evaluación del sistema mediante los interfaces orales: Seleccione la opción del menú “Asistente por voz” para probar únicamente el acceso al asistente mediante los interfaces orales.

10. ¿Qué tal le ha entendido el sistema?

- Muy mal
- Mal
- Regular
- Bien
- Muy bien

11. ¿Cómo ha entendido los mensajes generados por el sistema?

- Muy mal
- Mal
- Regular
- Bien
- Muy bien

12. En su opinión la interacción fue...

- Muy lenta
- Lenta
- Adecuada
- Rápida
- Muy rápida

13. Establezca el nivel de dificultad del sistema en modo oral para usted.

- Muy difícil
- Difícil
- Normal
- Fácil Muy fácil
- Muy fácil

14. ¿Ha percibido errores durante su interacción con el sistema en modo oral?

- Sí, he percibido muchos errores lo que ha imposibilitado la interacción con el sistema
- Sí, he percibido bastantes errores lo que ha causado gran dificultad en la interacción con el sistema
- Sí, he percibido algunos errores lo que ha causado cierta dificultad en la interacción con el sistema
- Sí, he percibido algunos errores aunque la interacción con el sistema no se ha visto afectada en absoluto
- No he percibido errores

15. ¿Le ha sido fácil decidir que hacer en cada momento para obtener la información que usted solicitaba?

- No, ha sido imposible
- No, me ha supuesto gran dificultad
- Sí, pero con ciertas dificultades
- Sí, ha sido fácil
- Sí, ha sido muy fácil

16. En términos generales, ¿está usted satisfecho con el sistema en modo oral?

- No, nada
- Poco satisfecho
- Satisfecho
- Bastante satisfecho
- Muy satisfecho

Pruebe ahora la comunicación multimodal con el sistema utilizando diferentes modalidades de entrada y/salida. Para ello, acceda las opciones del menú “Asistente televisivo”, “Recomendación de películas”, “buscar películas” y “Cartelera” y combine la utilización de los interfaces tradicionales (i.e. pantalla y teclado) con los interfaces orales.

17. ¿Qué tal le ha entendido el sistema?
    - Muy mal
    - Mal
    - Regular
    - Bien
    - Muy bien
18. ¿Cómo ha entendido los mensajes generados por el sistema?
    - Muy mal
    - Mal
    - Regular
    - Bien
    - Muy bien
19. En su opinión la interacción fue...
    - Muy lenta
    - Lenta
    - Adecuada
    - Rápida
    - Muy rápida
20. Establezca el nivel de dificultad del sistema multimodal para usted.
    - Muy difícil
    - Difícil
    - Normal
    - Fácil Muy fácil
    - Muy fácil
21. ¿Ha percibido errores durante su interacción con el sistema multimodal?
    - Sí, he percibido muchos errores lo que ha imposibilitado la interacción con el sistema
    - Sí, he percibido bastantes errores lo que ha causado gran dificultad en la interacción con el sistema
    - Sí, he percibido algunos errores lo que ha causado cierta dificultad en la interacción con el sistema
    - Sí, he percibido algunos errores aunque la interacción con el sistema no se ha visto afectada en absoluto
    - No he percibido errores

22. ¿Le ha sido fácil decidir que hacer en cada momento para obtener la información que usted solicitaba?

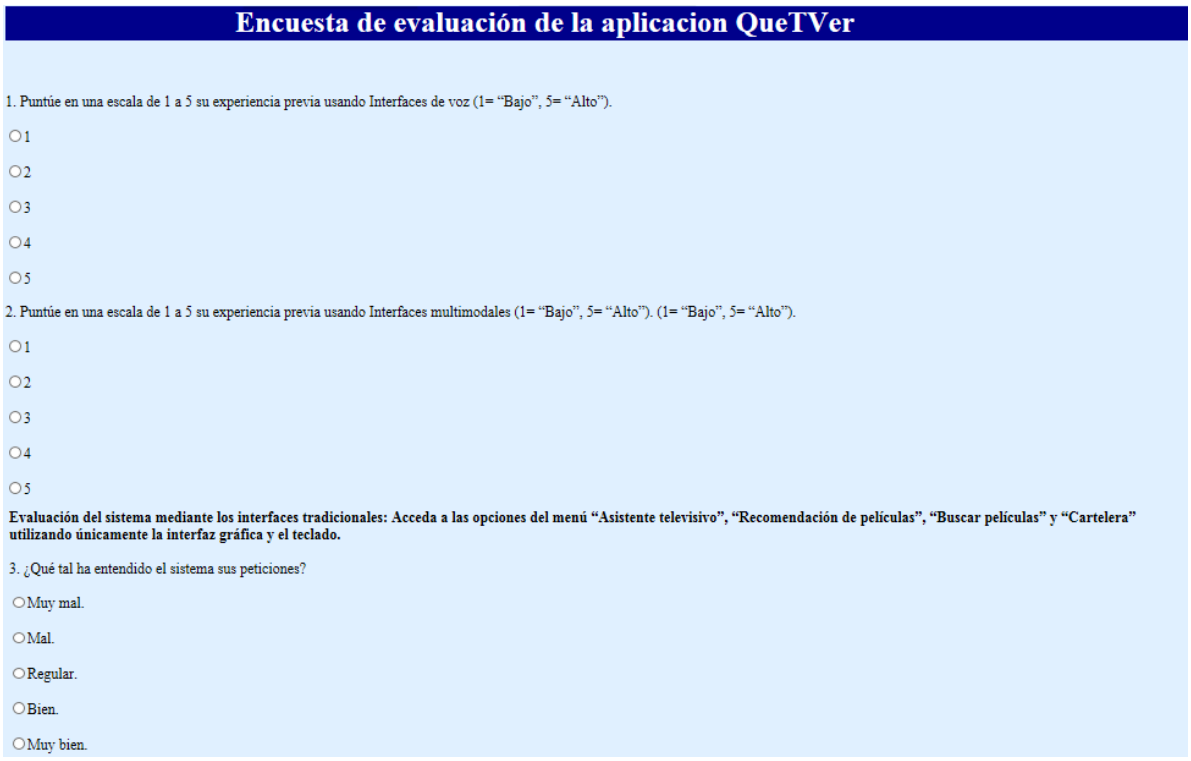
- No, ha sido imposible
- No, me ha supuesto gran dificultad
- Sí, pero con ciertas dificultades
- Sí, ha sido fácil
- Sí, ha sido muy fácil

23. En términos generales, ¿está usted satisfecho con el sistema multimodal?

- No, nada
- Poco satisfecho
- Satisfecho
- Bastante satisfecho
- Muy satisfecho

Figura 5.1: Cuestionario desarrollado para la evaluación subjetiva de la aplicación QueTVer

Para facilitar el registro de las opiniones de los usuarios, se ha desarrollado una página web (`http://proyectoquetver.x10.mx/encuesta/eval.php`) en la que se muestra el cuestionario tal y como se observa en la Figura 5.2.



4. ¿Cómo ha entendido los mensajes generados por el sistema?

- ○ Muy mal.
- ○ Mal.
- ○ Regular.
- ○ Bien.
- ○ Muy bien.

5. En su opinión la interacción fue…

- ○ Muy lenta.
- ○ Lenta.
- ○ Adecuada.
- ○ Rápida.
- ○ Muy rápida.

6. Establezca el nivel de dificultad del sistema en modo táctil para usted.

- ○ Muy difícil.
- ○ Difícil.
- ○ Normal.
- ○ Fácil.
- ○ Muy fácil.

7. ¿Ha percibido errores durante su interacción con el sistema en modo táctil?

- ○ Sí, he percibido muchos errores lo que ha imposibilitado la interacción con el sistema.
- ○ Sí, he percibido bastantes errores lo que ha causado gran dificultad en la interacción con el sistema.
- ○ Sí, he percibido algunos errores lo que ha causado cierta dificultad en la interacción con el sistema.
- ○ Sí, he percibido algunos errores aunque la interacción con el sistema no se ha visto afectada en absoluto.
- ○ No he percibido errores.

8. ¿Le ha sido fácil decidir que hacer en cada momento para obtener la información que usted solicitaba?

- ○ No, ha sido imposible.
- ○ No, me ha supuesto gran dificultad.
- ○ Sí, pero con ciertas dificultades.
- ○ Sí, ha sido fácil.
- ○ Sí, ha sido muy fácil.

9. En términos generales, ¿está usted satisfecho con el sistema en modo táctil?

- ○ No, nada.
- ○ Poco satisfecho.
- ○ Satisfecho.
- ○ Bastante satisfecho.
- ○ Muy satisfecho.

Evaluación del sistema mediante los interfaces orales: Seleccione la opción del menú "Asistente por voz" para probar únicamente el acceso al asistente mediante los interfaces orales.

10. ¿Qué tal ha entendido el sistema sus peticiones?

- ○ Muy mal.
- ○ Mal.
- ○ Regular.
- ○ Bien.
- ○ Muy bien.

11. ¿Cómo ha entendido los mensajes generados por el sistema?

- ○ Muy mal.
- ○ Mal.
- ○ Regular.
- ○ Bien.
- ○ Muy bien.

12. En su opinión la interacción fue…

- ○ Muy lenta.
- ○ Lenta.
- ○ Adecuada.
- ○ Rápida.
- ○ Muy rápida.

13. Establezca el nivel de dificultad del sistema en modo oral para usted.

- ○ Muy difícil.
- ○ Difícil.
- ○ Normal.
- ○ Fácil.
- ○ Muy fácil.

14. ¿Ha percibido errores durante su interacción con el sistema en modo oral?

- ○ Sí, he percibido muchos errores lo que ha imposibilitado la interacción con el sistema.
- ○ Sí, he percibido bastantes errores lo que ha causado gran dificultad en la interacción con el sistema.
- ○ Sí, he percibido algunos errores lo que ha causado cierta dificultad en la interacción con el sistema.
- ○ Sí, he percibido algunos errores aunque la interacción con el sistema no se ha visto afectada en absoluto.
- ○ No he percibido errores.

15. ¿Le ha sido fácil decidir que hacer en cada momento para obtener la información que usted solicitaba?

- ○ No, ha sido imposible.
- ○ No, me ha supuesto gran dificultad.
- ○ Sí, pero con ciertas dificultades.
- ○ Sí, ha sido fácil.
- ○ Sí, ha sido muy fácil.

16. En términos generales, ¿está usted satisfecho con el sistema en modo oral?

- ○ No, nada.
- ○ Poco satisfecho.
- ○ Satisfecho.
- ○ Bastante satisfecho.
- ○ Muy satisfecho.

Pruebe ahora la comunicación multimodal con el sistema utilizando diferentes modalidades de entrada y/salida. Para ello, acceda las opciones del menú "Asistente televisivo", "Recomendación de películas", "buscar películas" y "Cartelera" y combine la utilización de los interfaces tradicionales (i.e. pantalla y teclado) con los interfaces orales.

17. ¿Qué tal ha entendido el sistema sus peticiones?

- ○ Muy mal.
- ○ Mal.
- ○ Regular.
- ○ Bien.
- ○ Muy bien.

18. ¿Cómo ha entendido los mensajes generados por el sistema?

- ○ Muy mal.
- ○ Mal.
- ○ Regular.
- ○ Bien.
- ○ Muy bien.

19. En su opinión la interacción fue…

- ○ Muy lenta.
- ○ Lenta.
- ○ Adecuada.
- ○ Rápida.
- ○ Muy rápida.

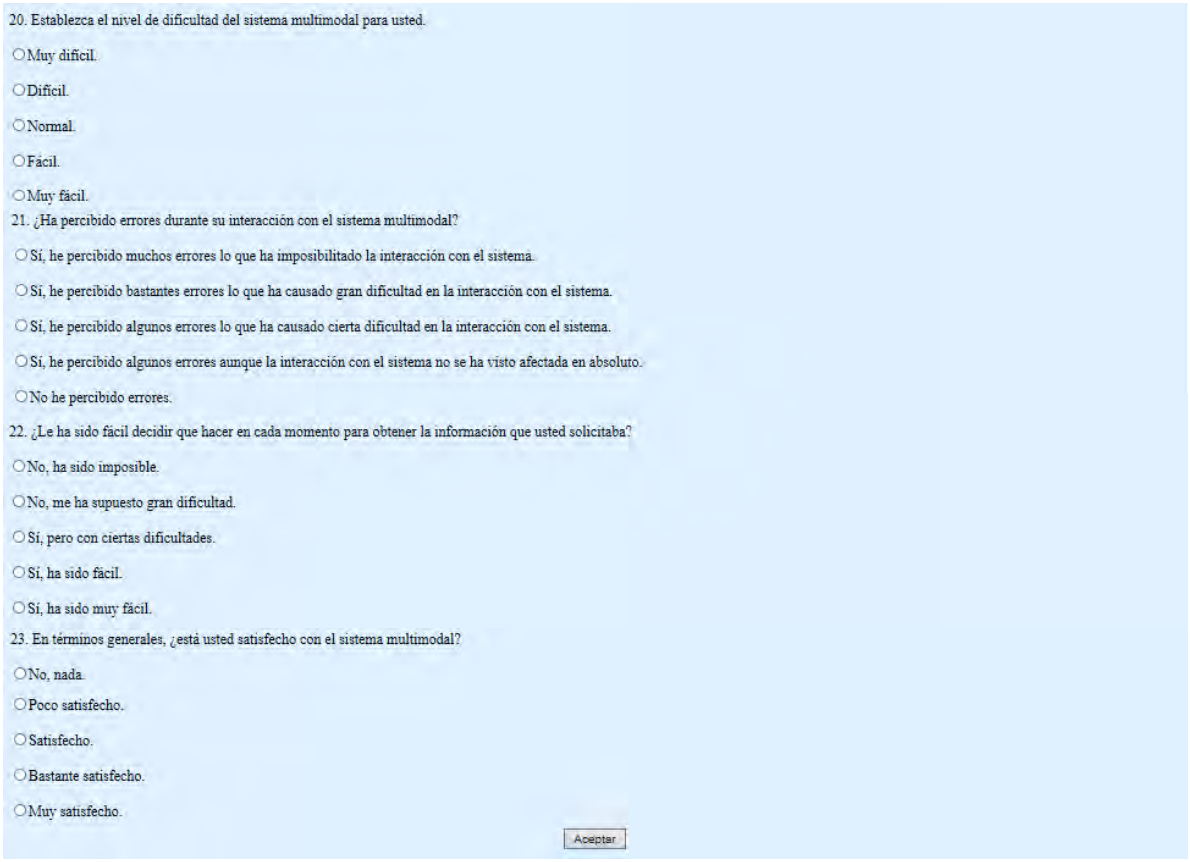
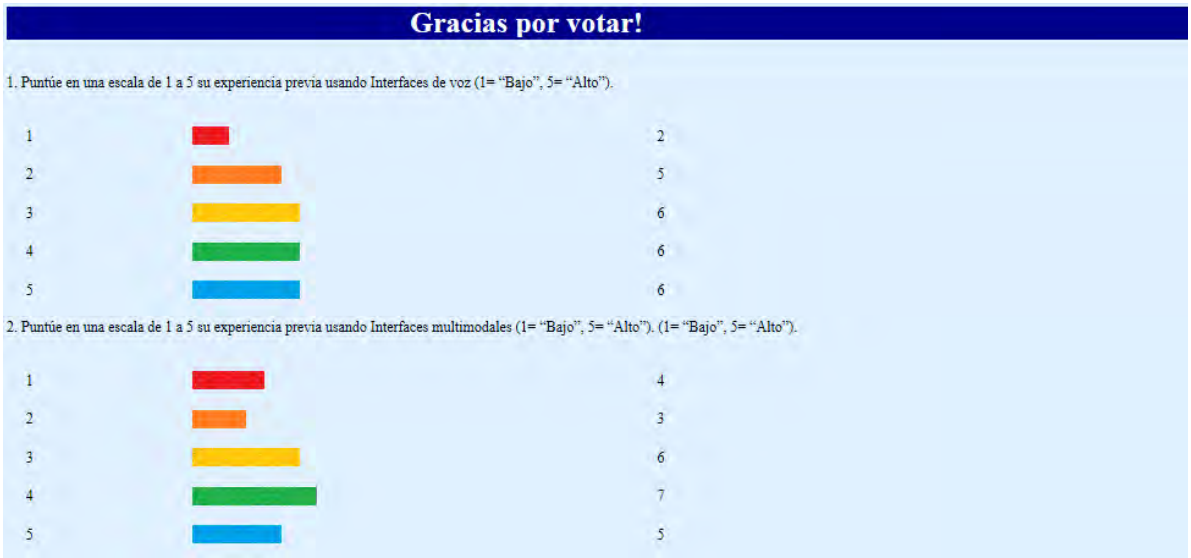


Figura 5.2: Página web para realizar la evaluación de la aplicación QueTVer

Una vez que el usuario rellena el cuestionario y pulsa el botón Aceptar, es dirigido a otra página web en el que se le da las gracias por participar y se muestran los resultados parciales de la encuesta hasta el momento, tal y como se muestra en la Figura 5.3.

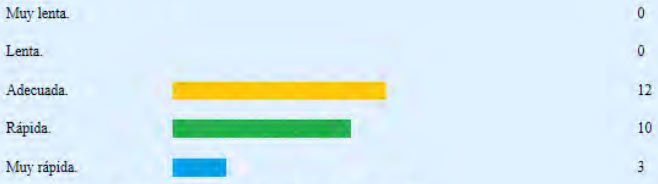


Evaluación del sistema mediante los interfaces tradicionales: Acceda a las opciones del menú "Asistente televisivo", "Recomendación de películas", "Buscar películas" y "Cartelera" utilizando únicamente la interfaz gráfica y el teclado.

3. ¿Qué tal ha entendido el sistema sus peticiones?

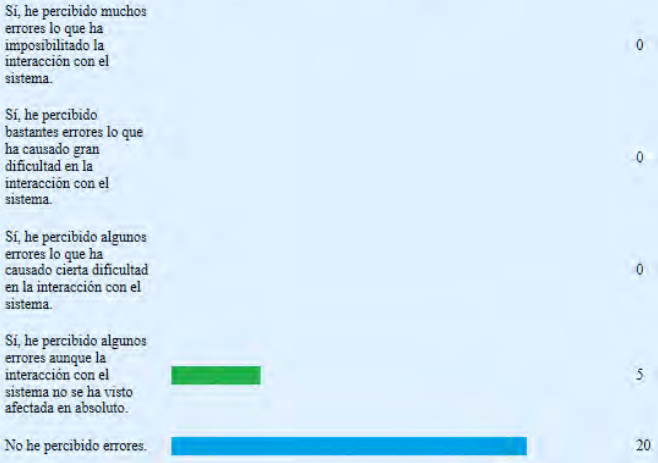
4. ¿Cómo ha entendido los mensajes generados por el sistema?

5. En su opinión la interacción fue…



6. Establezca el nivel de dificultad del sistema en modo táctil para usted.

7. ¿Ha percibido errores durante su interacción con el sistema en modo táctil?



8. ¿Le ha sido fácil decidir que hacer en cada momento para obtener la información que usted solicitaba?

No, ha sido imposible.	0
No, me ha supuesto gran dificultad.	0
Sí, pero con ciertas dificultades.	3
Sí, ha sido fácil.	19
Sí, ha sido muy fácil.	3

9. En términos generales, ¿está usted satisfecho con el sistema en modo táctil?

No, nada.	0
Poco satisfecho.	0
Satisfecho.	0
Bastante satisfecho.	14
Muy satisfecho.	11

Evaluación del sistema mediante los interfaces orales: Seleccione la opción del menú "Asistente por voz" para probar únicamente el acceso al asistente mediante los interfaces orales.

10. ¿Qué tal ha entendido el sistema sus peticiones?

Muy mal.	0
Mal.	0
Regular.	0
Bien.	9
Muy bien.	16

11. ¿Cómo ha entendido los mensajes generados por el sistema?

Muy mal.	0
Mal.	0
Regular.	9
Bien.	12
Muy bien.	4

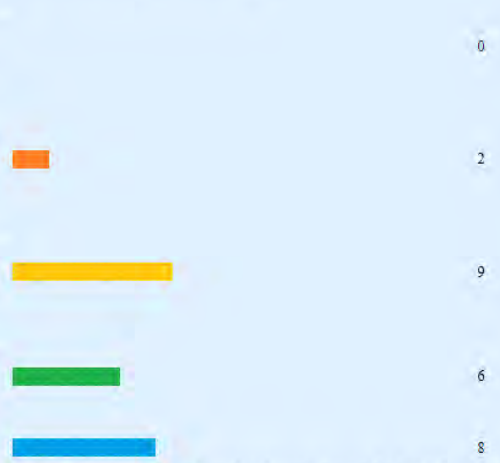
12. En su opinión la interacción fue…

Muy lenta.	0
Lenta.	13
Adecuada.	7
Rápida.	2
Muy rápida.	3

13. Establezca el nivel de dificultad del sistema en modo oral para usted.

Muy difícil.	0
Difícil.	0
Normal.	13
Fácil.	9
Muy fácil.	3

14. ¿Ha percibido errores durante su interacción con el sistema en modo oral?



15. ¿Le ha sido fácil decidir que hacer en cada momento para obtener la información que usted solicitaba?

16. En términos generales, ¿está usted satisfecho con el sistema en modo oral?

Pruebe ahora la comunicación multimodal con el sistema utilizando diferentes modalidades de entrada y/salida. Para ello, acceda las opciones del menú "Asistente televisivo", "Recomendación de películas", "buscar películas" y "Cartelera" y combine la utilización de los interfaces tradicionales (i.e. pantalla y teclado) con los interfaces orales.

17. ¿Qué tal ha entendido el sistema sus peticiones?

18. ¿Cómo ha entendido los mensajes generados por el sistema?

Figura 5.3: Página web con los resultados parciales de la evaluación de la aplicación QueTVer

Los votos de cada respuesta se almacenan automáticamente en la tabla *evaluacion* de la base de datos *proyec36_questionario*, de la que se han obtenido las estadísticas para el análisis de los resultados de la evaluación que se muestran en la siguiente sección de la memoria.

5.2. Resultados de la evaluación y conclusiones

El cuestionario recoge el grado de satisfacción de 25 usuarios hispanoparlantes (12 hombres, 13 mujeres) cuya edad comprende el rango de 22 a 54 años.

En primer lugar, los usuarios evaluaron el sistema utilizando el modo táctil. Posteriormente, evaluaron el sistema utilizando los interfaces orales. Finalmente, evaluaron el sistema multimodal combinando la utilización de los interfaces tradicionales (pantalla y teclado) con los interfaces orales. Los usuarios eligieron libremente las acciones a realizar durante la evaluación de la aplicación por lo que los módulos y sub-módulos a los que accedieron fueron variados.

Las Figuras 5.4 y 5.5 muestran las estadísticas correspondientes a las preguntas que evalúan la experiencia de los usuarios utilizando interfaces de voz y multimodales respectivamente. Las Figuras 5.6-5.12 comparan las estadísticas obtenidas en las tres modalidades de acceso a la aplicación para cada uno de los criterios evaluados: el grado en el cual el usuario valora que es entendido por el sistema, el grado en el cual el usuario valora que entiende los mensajes generados por el sistema, la velocidad percibida de la de interacción, el nivel de dificultad del sistema, la presencia de errores, la seguridad del usuario en lo que debe hacer en cada momento, y el nivel de satisfacción con el sistema global.

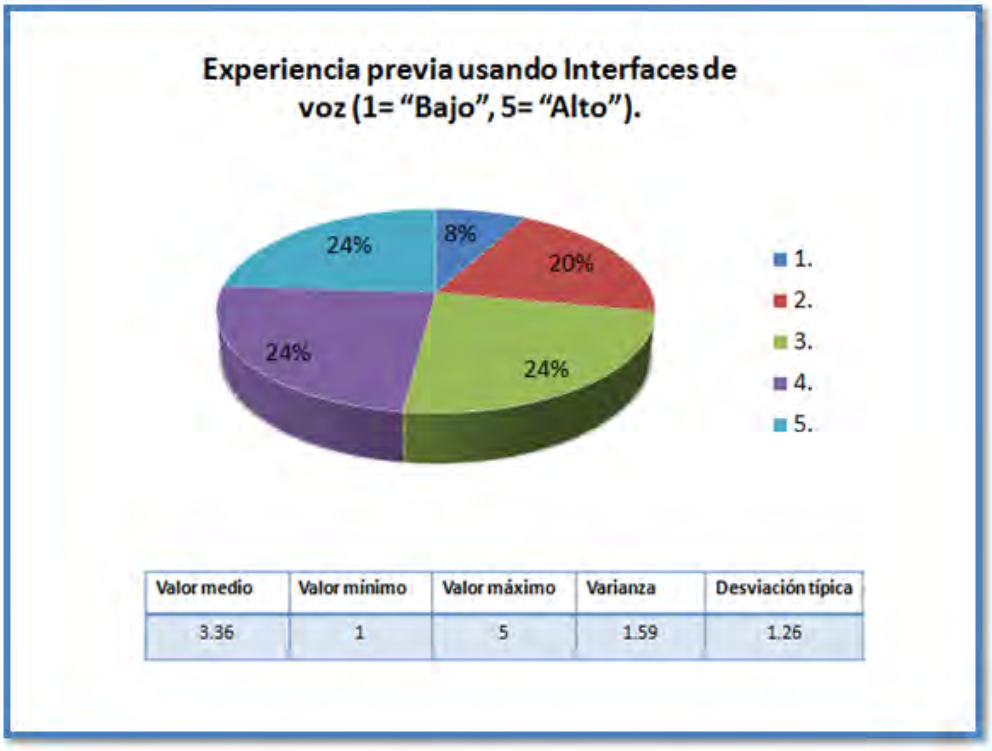
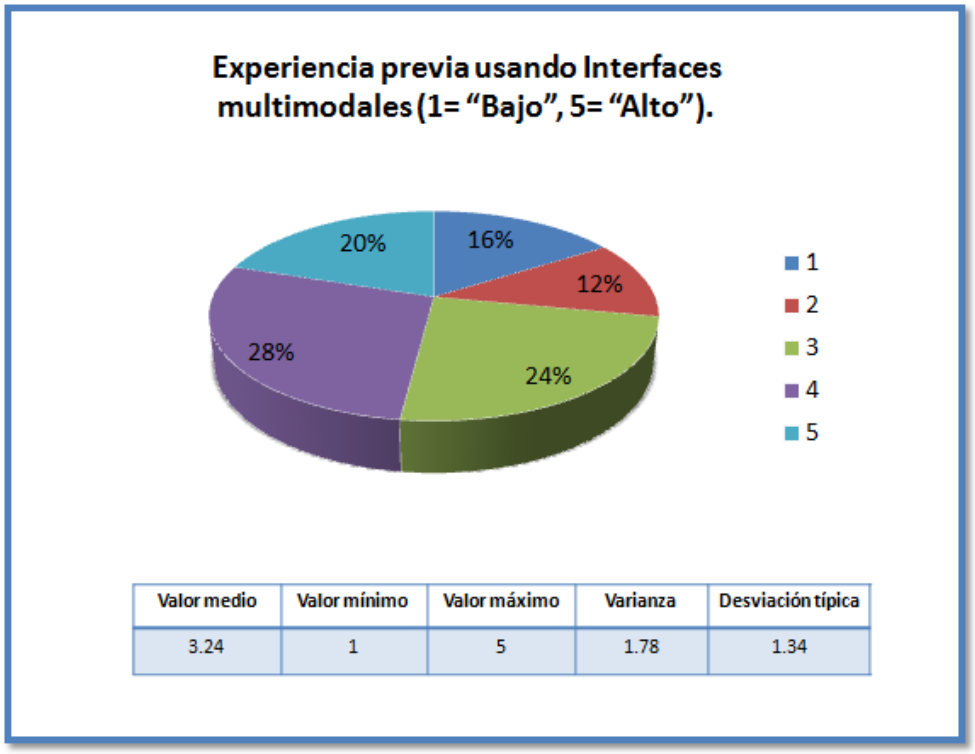
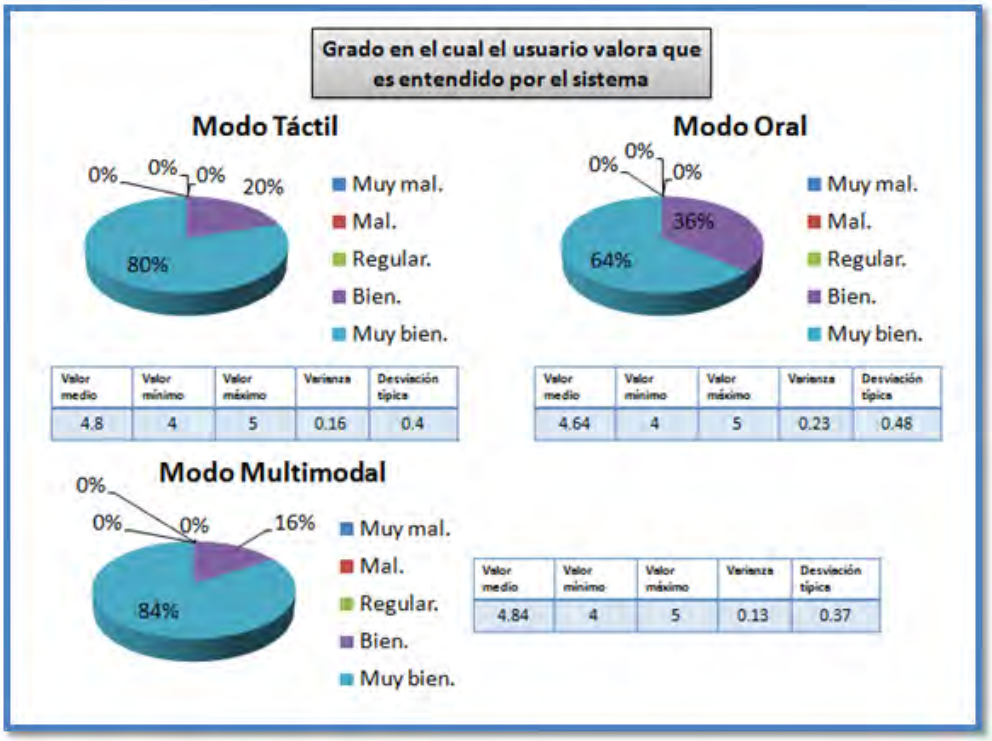


Figura 5.4: Estadísticas de la experiencia previa usando interfaces de voz



Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
3.24	1	5	1.78	1.34

Figura 5.5: Estadísticas de la experiencia previa usando interfaces multimodales



Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
4.8	4	5	0.16	0.4

Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
4.64	4	5	0.23	0.48

Valor medio	Valor mínimo	Valor máximo	Varianza	Desviación típica
4.84	4	5	0.13	0.37

Figura 5.6: Estadísticas del grado en el cual el usuario valora que es entendido por el sistema

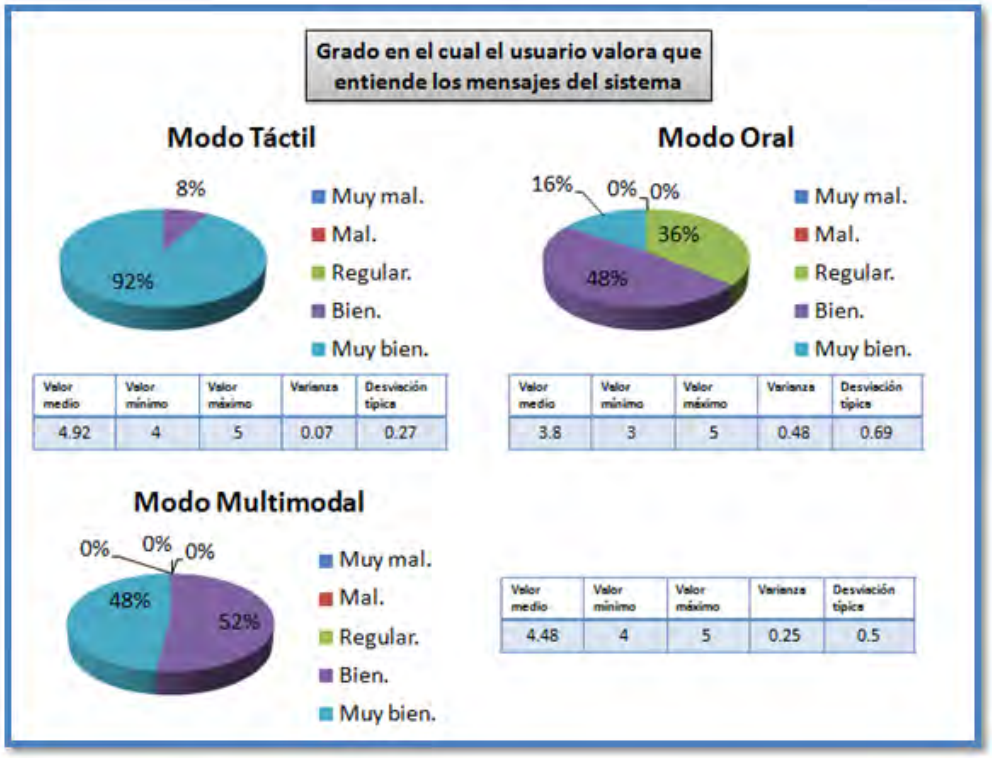


Figura 5.7: Estadísticas del grado en el cual el usuario valora que entiende los mensajes generados por el sistema

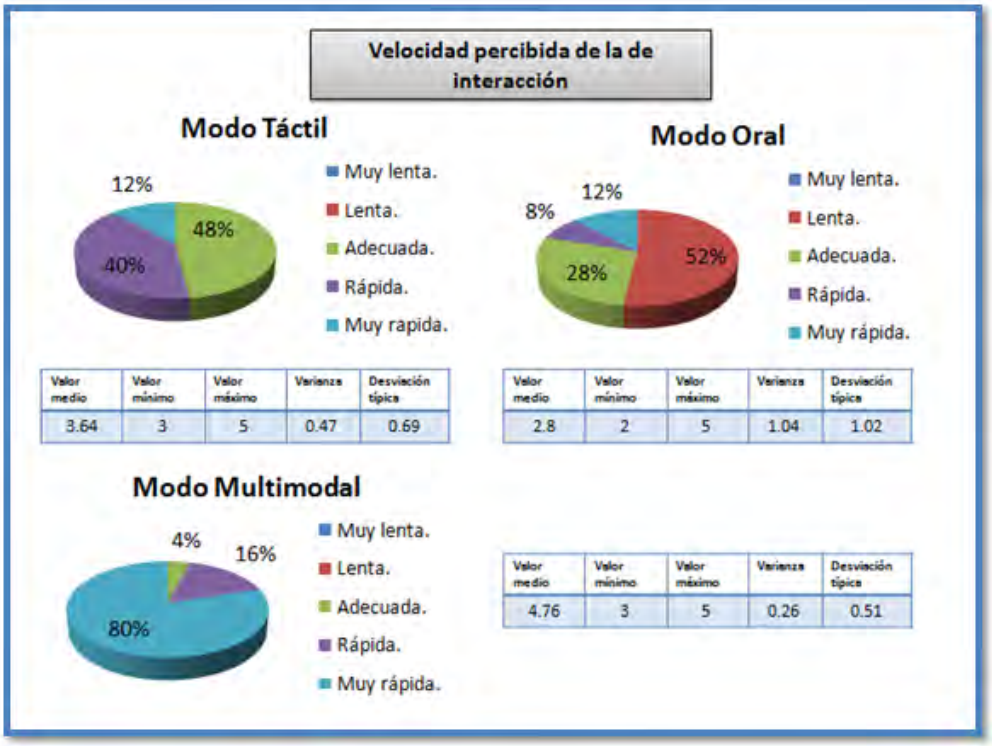


Figura 5.8: Estadísticas de la velocidad percibida de la interacción

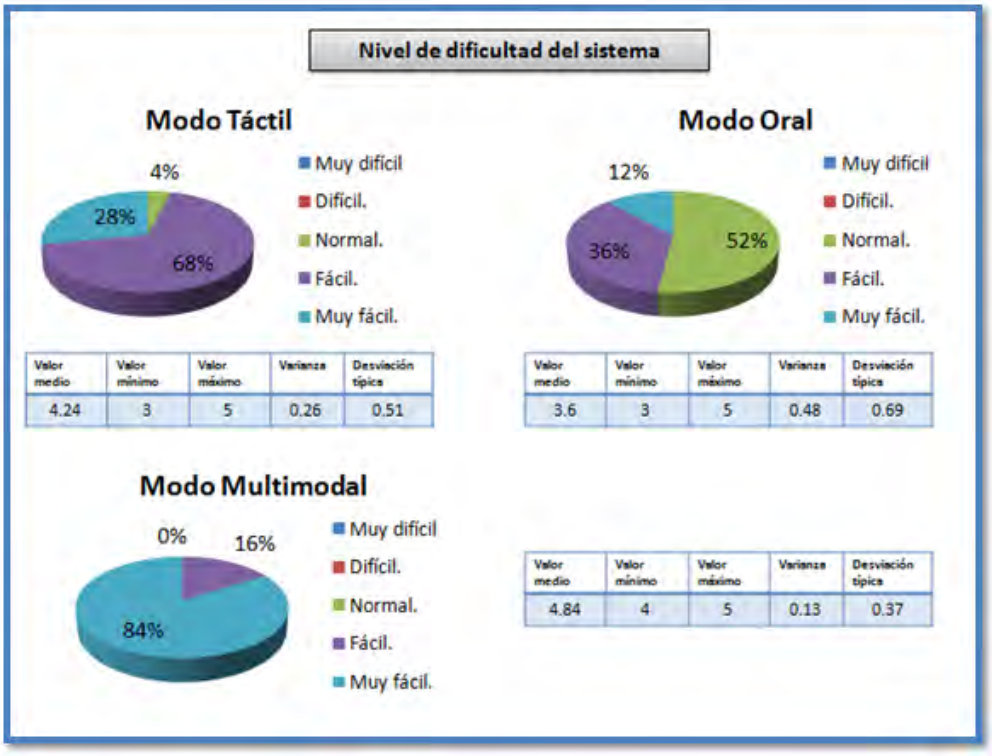


Figura 5.9: Estadísticas del nivel de dificultad del sistema

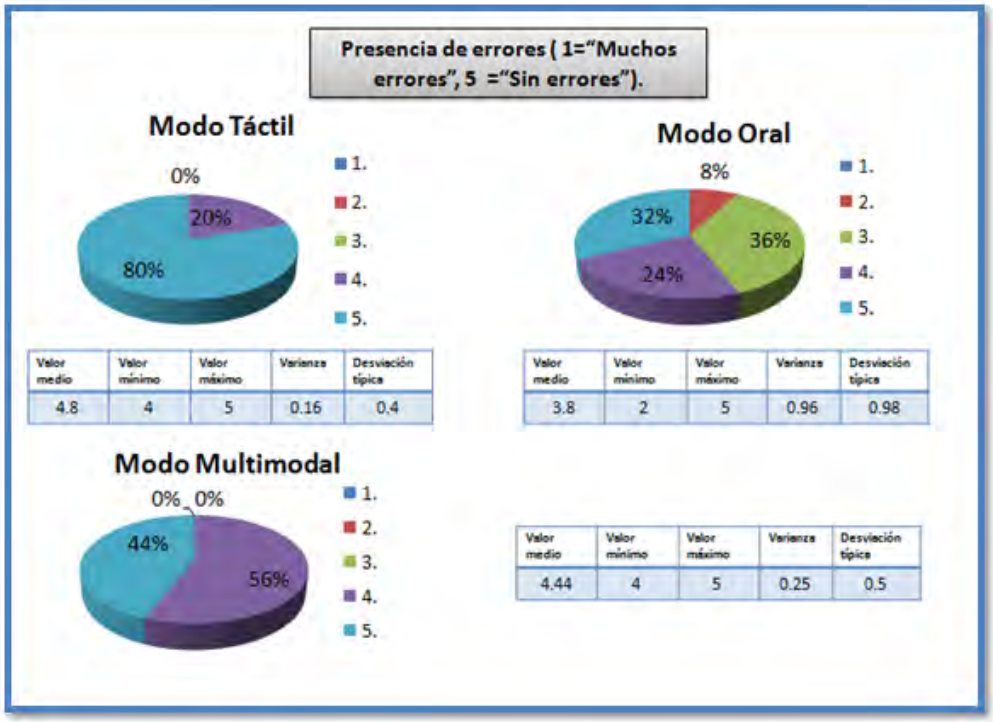


Figura 5.10: Estadísticas de la presencia de errores

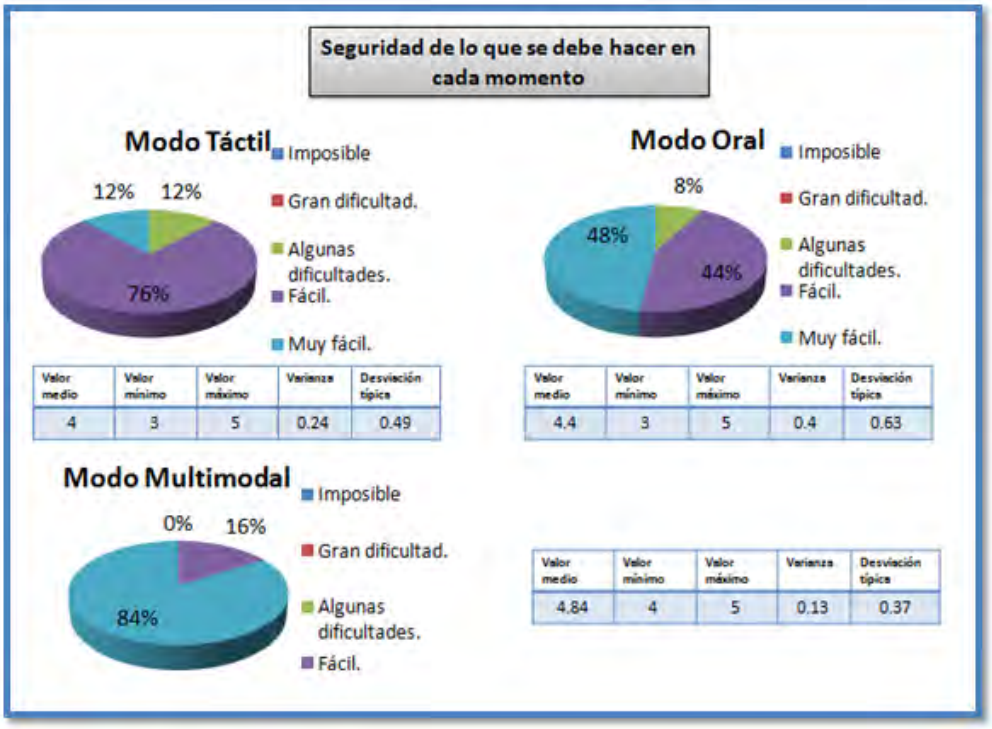


Figura 5.11: Estadísticas de la seguridad del usuario en lo que debe hacer en cada momento

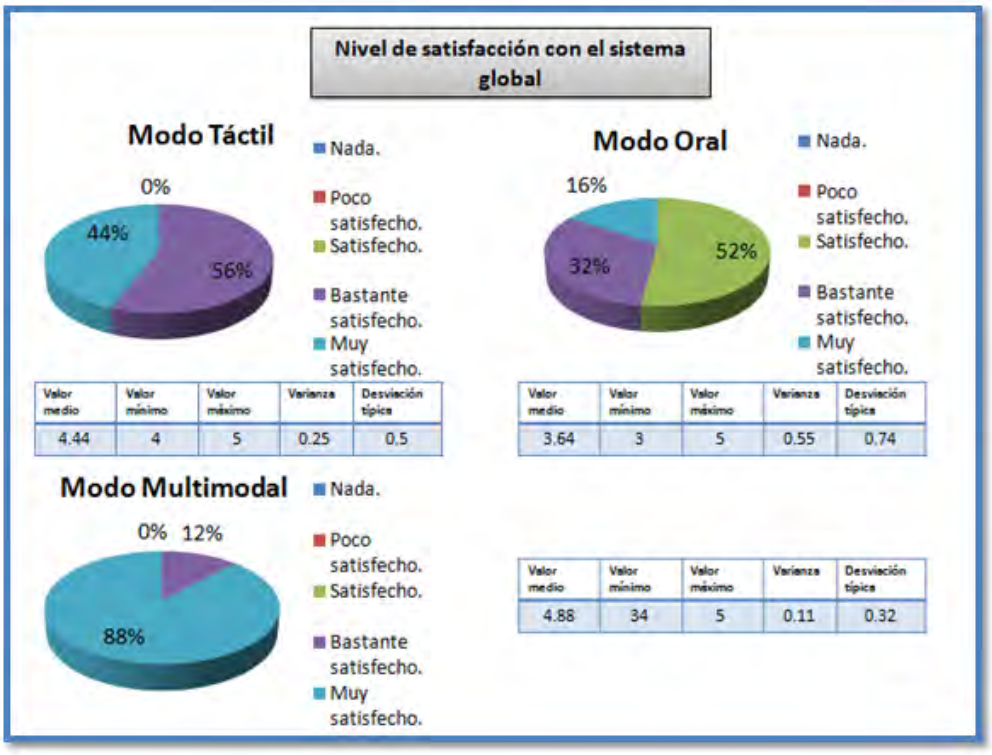


Figura 5.12: Estadísticas del nivel de satisfacción con el sistema global

En cuanto a los resultados obtenidos en las preguntas referidas a la evaluación del nivel de experiencia previa de los usuarios con interfaces orales y multimodales mostrados en la Figura 5.4 y Figura 5.5 respectivamente, se concluye que el nivel de conocimiento de los usuarios a los que se les ha realizado la encuesta es variado, siendo mayor el número de usuarios que poseen un conocimiento medio en la utilización de ambos modos de acceso. Se ha intentado seleccionar tanto a usuario familiarizados con los interfaces multimodales y orales como a usuarios que no tenían experiencia previa con el objetivo de extraer conclusiones más fiables.

Mediante el análisis de los resultados mostrados en las gráficas de las Figuras 5.6-5.12, se extraen las siguientes conclusiones:

Grado en el cual el usuario valora que es entendido por el sistema.

Para el modo táctil se esperaba obtener estos resultados de antemano. Para el modo oral y el modo multimodal estos resultados han sido posibles gracias a la eficacia del sistema de reconocimiento de voz de Google, que está en continua evolución y es capaz en la actualidad, de reconocer perfectamente cualquier secuencia de voz que el usuario emita. Además, para mejorar esta característica en el módulo ***Búsqueda de películas***, se ha tenido en cuenta que el título de algunas películas no está traducido, por lo que se permite al usuario pronunciar al el título de la película en su idioma original, seleccionando por adelantado el idioma en el cual pronunciará dicho título.

Grado en el cual el usuario valora que entiende los mensajes generados por el sistema.

Antes de la evaluación de esta característica, se esperaba que el modo multimodal obtuviese los mejores resultados de la encuesta ya que permite combinar el modo táctil con el oral y por lo tanto aprovechar las ventajas de ambos. Sin embargo, a pesar de que los usuarios hayan valorado en general que entienden muy bien los mensajes generados por el sistema en el modo multimodal, el modo que ha obtenido los mejores resultados ha sido el modo táctil, seguido del modo multimodal y finalmente del modo oral, siendo este último el que ha obtenido los peores resultados.

En cuanto al modo táctil no se esperaba obtener tan buenos resultados, ya que se presuponía de antemano que los usuarios con problemas de visión iban a tener mayores dificultades para leer la información en la pantalla del dispositivo y que les fuese más cómodo por lo tanto utilizar el modo oral o el modo multimodal. Sin embargo, al diseñar la aplicación se intentó que los resultados se mostrasen de forma clara por pantalla, utilizando un tamaño de fuente grande, por lo que todos los usuarios han entendido a la perfección los mensajes que ha generado el sistema.

El modo oral ha obtenido resultados satisfactorios aunque peores que los esperados. Los resultados en el modo oral se han visto afectados negativamente por dos factores:

- El grado en el cual usuario valora que entiende los mensajes generados por el sistema depende de la calidad de las voces que el usuario tenga instaladas en el dispositivo. Tal y como se ha visto en el estudio efectuado al principio de este proyecto, el sintetizador de voz IVONA TTS HQ es el que ofrece las voces de mayor calidad. Sin embargo, algunos usuarios no han optado por descargarse sus voces y han mantenido la voz que les viene por defecto en su dispositivo o la que pide instalar Google por defecto si no encuentra ninguna voz en la aplicación.

- Cuando el sintetizador está leyendo un texto en un determinado idioma, no diferencia las cadenas dentro del texto que están en otro idioma por lo que las pronuncia en el idioma en el que se estaba efectuando la síntesis de texto a voz. Por ejemplo, en el módulo ***Búsqueda de películas***, cuando el sintetizador lee la sinopsis de una determinada película en español y aparece en el texto el nombre de sus personajes en inglés, el sintetizador los pronuncia con acento español lo que dificulta el entendimiento de los mensajes generados por el sistema. Esto se podría solucionar ya que el diseñador puede especificar la pronunciación de determinadas cadenas de texto. Sin embargo, es muy difícil tener en cuenta todas las posibles cadenas existentes y el texto podría escucharse entrecortado. El módulo ***Recomendación de películas*** también se ve afectado por el problema de pronunciación en un idioma incorrecto ya que el sintetizador pronuncia toda la lista de películas en español aun habiendo películas cuyo título no está traducido. No obstante,

en el módulo ***Búsqueda de películas***, el sintetizador de voz lee el título original, reparto y el director de una determinada película en el idioma del país de origen de la película, evitando este problema.

Velocidad percibida de la interacción.

Los usuarios han valorado que la velocidad de la interacción en el modo multimodal ha sido muy rápida, en el modo táctil ha sido buena y en el modo oral ha sido adecuada.

Cabe destacar que la velocidad en interacción oral entre el usuario y la aplicación se ha visto dificultada al no existir un mecanismo en Android para que la síntesis de voz se detenga cuando el usuario desee comenzar a hablar. Esto resulta incómodo para el usuario considerando que este va aprendiendo cómo funciona la aplicación según va utilizándola, y no le resulta necesario esperar a que termine de hablar la aplicación para emitirle sus peticiones a través del reconocedor de voz.

El modo multimodal ha obtenido los mejores resultados para esta característica ya que el usuario puede seleccionar la modalidad que le resulte más cómoda y rápida de utilizar en todo momento. Además, puede pulsar un botón para pausar la síntesis y un botón para iniciar el reconocimiento de voz, evitando por lo tanto la desventaja principal del modo oral.

Nivel de dificultad del sistema.

Los usuarios han valorado que la utilización de la aplicación en modo multimodal ha sido la más fácil de utilizar. También han valorado la interacción con la aplicación en modo táctil como muy fácil y la interacción en modo oral como fácil.

El modo multimodal ha obtenido los mejores resultados para esta característica ya que el usuario puede seleccionar la modalidad que le resulte más cómoda y fácil de utilizar en todo momento, sin estar restringido a las dificultades que pueda encontrar en un único modo y evitar de esta forma quedarse atascado sin saber qué decisión tomar.

En el modo multimodal y oral se emiten por voz instrucciones de utilización de la aplicación lo que facilita la interacción con la misma. El modo táctil no muestra por pantalla las instrucciones de utilización por lo que el usuario ha de apoyarse únicamente en su intuición a la hora de decidir qué decisión tomar.

Presencia de errores.

Los usuarios no han percibido en general errores en su interacción con la aplicación en modo táctil. En el modo multimodal y oral han percibido algunos errores pero que no han dificultado en absoluto su interacción con la aplicación. Aun así, el modo oral es el que se ha visto más afectado ante la presencia de errores ya que el usuario puede sentirse más inseguro a la hora de tomar una decisión después de la ocurrencia de un error al no haber aun un mecanismo de *feedback* lo suficientemente logrado en el modo oral. Por lo general, el usuario valora que es más fácil apoyarse en lo que ve por pantalla cuando ha sucedido algún error.

Seguridad del usuario en lo que debe hacer en cada momento.

Los usuarios han sentido una mayor seguridad en lo que debían hacer en cada momento con la interacción en modo multimodal, seguido de la interacción en modo oral y finalmente de la interacción en modo táctil. A pesar de ello, ha sido fácil tomar decisiones en los tres modos de acceso a la aplicación.

El motivo principal por el cual el modo multimodal y el modo oral resultan más ventajoso en comparativa con el modo táctil en cuanto a esta característica es que, tal y como se ha explicado anteriormente, la aplicación en estos modos emite instrucciones de utilización de la aplicación y cuenta con más mecanismos de *feedback*. El modo táctil se apoya únicamente en la intuición que tiene el usuario a la hora de decidir qué acción tomar. Aun así, el usuario se siente más cómodo utilizando

el modo multimodal que el modo oral ya que si no entiende alguna instrucción u ocurre algún error puede apoyarse también en lo que ve por pantalla.

Nivel de satisfacción con el sistema global.

Los usuarios de la aplicación encuestados se han sentido muy satisfechos con el modo multimodal y táctil, y satisfechos con el modo oral.

El modo multimodal ha sido el mejor valorado por los usuarios ya que como se desprende de las conclusiones extraídas del resto de preguntas de la encuesta, el usuario prefiere combinar el modo táctil y el modo oral según le resulte más eficaz, aprovechando todas las ventajas y evitando todas las desventajas de ambos modos.

Capítulo 6

CONCLUSIONES Y TRABAJO FUTURO

En este capítulo se realiza una revisión global del proyecto y se exponen las conclusiones finales extraidas a partir de los objetivos marcados inicialmente. Además, se explican las dificultades encontradas durante el desarrollo del proyecto y se presentan algunas sugerencias de mejora para el desarrollo de aplicaciones para dispositivos móviles Android con interfaces orales. Finalmente, se describen posibles lineas de trabajo que se podrían generar a partir de este proyecto con el fin de mejorar la aplicación desarrollada.

6.1. Conclusiones

En el presente Proyecto Final de Carrera se ha desarrollado una aplicación multimodal para dispositivos Android cumpliendo de esta forma con el objetivo principal del proyecto. Se trata de la aplicación denominada QueTVer, cuya función es la de proporcionar al usuario, a través de un sistema de diálogo o través de los interfaces tradicionales como el teclado o la pantalla, información y recomendaciones de televisión y de cine atendiendo a los gustos registrados por el usuario.

Los servicios ofrecidos por la aplicación QueTVer se reparten en módulos a los que el usuario puede acceder interactuando con el sistema de varias formas:

- **Interacción oral:** La opción ***Asistente por voz*** del menú principal conduce al usuario a los distintos módulos de la aplicación en función de las decisiones que vaya tomando en cada diálogo.
- **Interacción multimodal:** El resto de las opciones del menú principal permiten al usuario tener acceso multimodal a los distintos módulos del sistema. En todo momento, la aplicación permite al usuario seleccionar la modalidad con la que desea interactuar con el dispositivo de acuerdo a sus preferencias y a las condiciones del ambiente de uso. El usuario puede interactuar con el sistema mediante el modo táctil, el modo oral, o combinar ambos modos.

A continuación, se realiza un repaso de los servicios que ofrece la aplicación QueTVer una vez que ha sido desarrollada.

- **Asistente televisivo.**

  El módulo ***Asistente televisivo*** recomienda la programación diaria emitida en televisión de acuerdo a las preferencias televisivas que registra el usuario en un cuestionario: deportes, cine, documentales, música, noticias, series de televisión, programas de televisión, realities y/o concursos.

  En primer lugar, el usuario debe actualizar la base de datos con la programación del día de la consulta en el caso de no haberlo aún durante el día actual. En el caso contrario, no es necesario este primer paso. Posteriormente, el usuario selecciona el canal de televisión sobre el que desea

conocer la programación correspondiente a sus gustos. Los canales televisivos sobre los que el sistema tiene datos son: TVE La Primera, TVE La 2, Antena 3, Cuatro, Tele 5, La Sexta, TeleMadrid, La Sexta3 y Paramount Channel. A continuación, el usuario selecciona la categoría televisiva sobre la que desea obtener la información entre las categorías televisivas coincidentes con las que el sistema ha encontrado según la programación del día actual y con las que el usuario registró en el cuestionario. Finalmente, el asistente televisivo devuelve la programación televisiva correspondiente al canal y la categoría seleccionada por el usuario.

El usuario puede consultar información más específica acerca de las películas, documentales o series de televisión recomendadas ya que el módulo ***Asistente televisivo*** dirige al usuario al módulo ***Búsqueda de películas*** en el caso de que este lo requiera. Esta función aún no está disponible para la opción ***Asistente por voz*** del menú principal que permite al usuario interactuar con la aplicación únicamente de forma oral.

El presente módulo requiere tener acceso a Internet una vez al día para actualizar la base de datos con la programación del día de la consulta y en el caso de que el usuario quiera utilizar el reconocimiento de voz para interactuar con el sistema. Para el resto de las funcionalidades ofrecidas por el módulo no es necesario disponer de acceso a Internet.

- **Recomendación de películas.**

  El módulo ***Recomendación de películas*** recomienda películas, series de TV y documentales de acuerdo a los gustos que registra el usuario en un cuestionario: género, país y rango de fechas. Además, el usuario puede decidir si excluir de los resultados devueltos por el módulo las series de TV y/o los documentales.

  A medida que el presente módulo devuelve las recomendaciones al usuario, este puede o bien solicitar que se le recomiende la siguiente opción, o solicitar información más específica acerca de la película, documental o serie de televisión recomendada ya que el módulo ***Recomendación de películas*** dirige al usuario al módulo ***Búsqueda de películas*** en el caso de que este lo requiera. Esta función aún no está disponible para la opción ***Asistente por voz*** del menú principal que permite al usuario interactuar con la aplicación únicamente de manera oral.

  El presente módulo no requiere tener acceso a Internet para ofrecer al usuario la funcionalidad completa.

- **Búsqueda de películas.**

  El módulo ***Búsqueda de películas*** permite al usuario solicitar información de una película (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios). El usuario puede seleccionar el idioma con el que va a pronunciar el título de la película en el caso de utilizar el reconocimiento de voz como entrada al módulo.

  El presente módulo requiere tener acceso a Internet para todas las funcionalidades que ofrece al usuario.

- **Cartelera.**

  El módulo ***Cartelera*** permite al usuario buscar salas de cine indicando una provincia y posteriormente seleccionar uno de los cines encontrados en dicha provincia y obtener la información de la cartelera correspondiente a dicho cine: películas en cartelera, duración de las películas en cartelera, horarios de emisión y clasificación por edades.

  El usuario puede consultar información más específica acerca de las películas en cartelera ya que el módulo ***Cartelera*** dirige al usuario al módulo *Búsqueda de películas* en el caso de que este lo requiera. Esta función aún no está disponible para la opción ***Asistente por voz*** del menú principal que permite al usuario interactuar con la aplicación únicamente de manera oral.

  El presente módulo requiere tener acceso a Internet para ofrecer toda su funcionalidad al usuario.

La aplicación QueTVer permite al usuario registrar el nombre de usuario, contraseña y sus gustos de cine y de televisión en una base de datos MySQL remota a través de un cuestionario. De esta forma,

en futuras interacciones con la aplicación, el usuario puede identificarse desde cualquier dispositivo Android y la aplicación tendrá acceso a los gustos televisivos y de cine del usuario sin la necesidad de que este vuelva a rellenar el formulario.

No obstante, la aplicación también ofrece al usuario la posibilidad de acceder a sus servicios sin la necesidad de identificarse. Esta opción no requiere tener acceso a Internet pero para que el usuario pueda acceder a los servicios ofrecidos por el módulo ***Asistente televisivo*** y por el módulo ***Recomendación de películas***, debe de registrar sus gustos de cine y de televisión a través de un cuestionario que en esta caso son enviados directamente a una base de datos SQLite interna.

Por lo tanto, cabe destacar que la aplicación puede funcionar en modo *offline* en el caso de no disponer de una conexión a Internet pero en este caso, únicamente se podrá aprovechar la funcionalidad ofrecida por los módulos ***Recomendación de películas*** y ***Asistente televisivo***, a pesar de que este último necesite Internet por lo menos una vez al día para poder actualizar la base de datos con la programación del día de la consulta. Además, no se podrá interactuar con el sistema a través del reconocimiento de voz. Por lo tanto, es importante disponer de acceso a Internet para poder aprovechar los servicios ofrecidos por la aplicación QueTVer en su totalidad.

Una vez concluida la revisión completa de la aplicación QueTVer, cuyo desarrollo constituía el objetivo principal del presente Proyecto Final de Carrera, se puede hacer un balance de los resultados obtenidos a partir de los objetivos parciales inicialmente marcados, que han ido necesarios para alcanzar el objetivo principal.

- **Estudio de los sistemas de diálogo.**

  En primer lugar, se ha realizado un estudio completo de los sistemas de diálogo que incluye una descripción de la funcionalidad, la arquitectura y de los módulos que los conforman; un análisis de los requisitos que debe cumplir un sistema de diálogo ideal y de las limitaciones a las que están sujetos; y algunos ejemplos existentes de los sistemas de diálogo en la actualidad. Este estudio ha sido necesario para poder incorporar a la aplicación desarrollada las características típicas de los sistemas de diálogo ideales y de este modo aprovechar toda la potencia que ofrecen dichos sistemas.

- **Estudio de la plataforma Android.**

  En cuanto a la plataforma Android, se ha realizado un estudio exhaustivo de su arquitectura, componentes, características y funcionamiento ya que se desconocía todo lo relacionado con dicha plataforma antes de realizar el presente Proyecto Final de Carrera. Este objetivo se ha cumplido gracias a la completa documentación que ofrece Google a los desarrolladores consistente mayoritariamente en un desglose detallado de los paquetes, clases e interfaces que componen el SDK de Android. Además, se han consultado foros, blogs y otras publicaciones en Internet proporcionadas por desarrolladores de este sistema.

  En base al estudio realizado de las características de Android y a la experiencia adquirida tras el desarrollo de la aplicación, se exponen algunos aspectos que resultan ventajosos en su elección como plataforma:

  - Al ser un sistema de **código abierto** cualquiera puede estudiar, modificar, mejorar y distribuir el sistema Android sin ningún tipo de restricción. La comunidad de desarrolladores es muy activa y está creando continuamente soluciones para Android. Además, Android ofrece la opción al desarrollo privado mediante la publicación comercial de aplicaciones, permitiendo a cada desarrollador decidir cómo quiere distribuir su propio trabajo.
  - A pesar de que Google evita utilizar el término demasiado, el que se utilice el **lenguaje de programación Java** para desarrollar las aplicaciones para dispositivos móviles Android ayuda a que cualquier programador que tenga una mínima experiencia con el lenguaje de programación Java pueda comenzar a programar aplicaciones para dispositivos móviles Android sin demasiada dificultad.
  - Cualquier aplicación Android es una **combinación de componentes** lo que ayuda a modularizar funcionalmente las aplicaciones.

- Android facilita el **diseño de interfaces** de usuario ya que incorpora un amplia variedad de elementos y además ofrece la posibilidad de definir la interfaz tanto en el código como a través de documentos XML externos. Declarar el diseño de interfaz de usuario en XML en lugar de utilizar código en tiempo de ejecución es útil ya que permite crear diferentes diseños para diferentes tamaños u orientaciones de la pantalla.
- El acceso a los **recursos del dispositivo** (Ej. Wi-Fi, servicio de reconocimiento de voz, motor de síntesis, base de datos SQLite, etc.) es una tarea sencilla gracias a las bibliotecas API que Android ofrece en su SDK ya que ayudan a que el desarrollador no tenga que programar a bajo nivel para que una aplicación pueda acceder a los componentes de hardware de los dispositivos.
- La plataforma Android ofrece un **entorno de desarrollo** completo que incluye un emulador de dispositivos, herramientas para la depuración, la memoria y perfiles de rendimiento, y un plug-in para el IDE de Eclipse con lo que se facilita enormemente la tarea de programación en este sistema. Así, pueden crearse proyectos completos para Android, incluyendo el manifiesto y la declaración de recursos externos, sin salir del entorno de desarrollo de Eclipse.

- **Análisis de las posibilidades que ofrece Android para el desarrollo de aplicaciones que permitan la interacción oral con el usuario.**

  Antes de comenzar con el desarrollo de la aplicación multimodal para dispositivos móviles Android, ha sido necesario realizar un análisis detallado de las posibilidades que ofrece la plataforma Android para integrar el reconocimiento de voz y la síntesis de texto a voz en una aplicación.

  Tal y como se ha visto, Google con Android ha potenciado desde sus inicios la forma de interactuar de forma oral con el sistema ofreciendo multitud de posibilidades al desarrollador para integrar el reconocimiento de voz y la síntesis de texto a voz en una aplicación. Para realizar un análisis completo de estas posibilidades se ha partido de la documentación oficial ofrecida por Google sobre los paquetes android.speech y android.speech.RecognizerIntent para el reconocimiento de voz y sobre el paquete android.speech.tts para la síntesis de texto a voz. Además, se han consultado libros, foros, blogs y otras publicaciones en Internet proporcionadas por desarrolladores de este sistema.

  Partiendo del análisis efectuado, se ha aplicado la mejor de todas las posibilidades en el desarrollo de la nueva aplicación, tanto en la integración del reconocimiento de voz para la entrada como en la integración de la síntesis de texto a voz en la salida.

  - En cuanto al **reconocimiento de voz**, se ha visto que la opción más simple es la de utilizar el servicio de voz ofrecido por Google ya que ofrece altas prestaciones además de tener una fácil integración en las aplicaciones. Por consiguiente, para integrar el reconocimiento de voz en la aplicación para dispositivos móviles Android, se ha utilizado la clase android.speech.RecognizerIntent del SDK de Android con el objetivo de enviar un *RecognizerIntent* a la aplicación de búsqueda por voz de Google, siendo necesaria su previa instalación en el dispositivo. Cabe destacar que para utilizar el servicio de reconocimiento de voz de Google es necesario tener acceso a Internet.
  - En cuanto a la **síntesis de texto a voz**, se ha visto el sistema operativo Android 1.6 o superior incorpora en la mayoría de sus dispositivos un motor de síntesis de voz denominado Pico TTS o Google TTS, el cual permite integrar de forma sencilla la síntesis de voz a cualquier aplicación mediante la utilización del paquete android.speech.tts perteneciente al SDK de Android, en particular mediante la utilización de su clase TextToSpeech. No obstante, Android ofrece la posibilidad de instalar y personalizar varios motores, aunque siempre eligiendo uno como motor de síntesis principal. A la hora de elegir el motor de síntesis utilizado en la aplicación se ha tenido en cuenta la calidad de las voces que ofrece, los idiomas que tiene disponibles para Android, el tamaño de la descarga, la disponibilidad y el precio. Tras el estudio de los motores existentes, se ha decidido emplear el motor de síntesis IVONA TTS HQ, ya que ofrece multitud de voces femeninas y masculinas cuya calidad es la mejor junto a la de las voces del motor SVOX Classic TTS, pero a diferencia

de este último, el motor IVONA TTS HQ es actualmente gratuito. Además, se encuentra disponible en Google Play por lo que se puede descargar de forma sencilla.

Además, con el objetivo de poder comenzar a perfilar la naturaleza de la aplicación a desarrollar, se han consultado ejemplos de asistentes virtuales y de otras aplicaciones de interés para dispositivos móviles Android existentes en la actualidad y similares a la aplicación que se ha desarrollado para este proyecto. La mayor parte de dichos asistentes de voz integran el reconocimiento de voz de Google para conectarse a los servicios de voz de Google aunque algunos de ellos utilizan servicios de voz propios. En cuanto a la síntesis de texto a voz, gran parte de dichos asistentes utilizan el motor TTS principal (especificado por el usuario) para realizar las lecturas de texto.

- **Estudio de otras tecnologías necesarias en el desarrollo de la aplicación para dispositivos móviles Android.**

  Además del estudio realizado de la plataforma Android y de la metodología de integración del reconocimiento de voz y síntesis de texto a voz en una aplicación Android, ha sido necesario realizar un estudio sobre la integración de las siguientes funcionalidades en una aplicación Android:

  - Crear, gestionar y manipular una base de datos SQLite en Android, necesario para almacenar los datos requeridos los módulos ***Asistente televisivo*** y ***Recomendación de películas***. Las razones por las que se utiliza SQLite como sistema de gestión de base de datos en estos módulos son:
    - SQLite es parte del SDK de Android y por tanto no es necesaria su descarga ni incluir librerías adicionales durante el desarrollo de la aplicación.
    - No es necesario disponer de acceso a Internet para poder consultar información de la base de datos SQLite por lo que la aplicación puede funcionar en modo *offline* para estos módulos.
    - SQLite realiza operaciones de manera eficiente y es más rápido que otros sistemas de gestión de base de datos como MySQL.
    - Los módulos ***Asistente televisivo*** y ***Recomendación de películas*** no necesitan almacenar una gran cantidad de datos por lo que SQLite es suficientemente grande para alcanzar los requisitos de almacenamiento de dichos módulos.

    Sin embargo, SQLite es una base de datos con funcionalidad limitada a comparación con otras como MySQL y por tanto su utilización se desaconseja en aplicaciones cliente-servidor, en sistemas en los que se desee utilizar bases de datos de tamaño muy grande y en aplicaciones que requieran alta concurrencia.
  - Desarrollar una arquitectura cliente-servidor de modo que la aplicación Android pueda acceder a una base de datos MySQL de un servidor web remoto**.** Esto se necesita para el almacenamiento de datos en el módulo ***Registro e identificación de usuario*** de forma que el usuario pueda iniciar sesión desde cualquier dispositivo mediante su nombre y contraseña y que la aplicación pueda acceder a su información de usuario. Para la parte web se ha utilizado el servidor web gratuito *x10hosting* que se encarga de almacenar los ficheros PHP y las bases de datos MySQL de la aplicación. Además, para facilitar la administración de las bases de datos se utiliza *PhpMyAdmin* ya que se trata de un completo y potente administrador de bases de datos MySQL que se utiliza vía web. En cuanto a la aplicación Android (cliente), se conecta y envía una consulta al servidor mediante HTTP y obtiene una respuesta en formato JSON, tratándose del formato ideal para el intercambio de datos al ser completamente independiente del lenguaje con el que se está programando.
  - Ejecución de tareas en segundo plano mediante la utilización de la clase AsyncTask proporcionada por Android. Esto es necesario ya que cualquier operación larga o costosa que se realice en el hilo principal, siendo este el hilo donde se ejecutan todas las operaciones que gestionan la interfaz de usuario de la aplicación, va a bloquear la ejecución del resto de componentes de la aplicación y de la interfaz, produciendo al usuario un efecto de lentitud, bloqueo, o mal funcionamiento. Además, dado que Android monitoriza las operaciones

realizadas en el hilo principal y detecta aquellas que superen los 5 segundos, mostrará el mensaje de *Application Not Responding* en el que el usuario debe decidir entre forzar el cierre de la aplicación o esperar a que termine. Para evitar este efecto, Android proporciona la clase auxiliar AsyncTask, que permite ejecutar tareas en segundo plano en Android. En la aplicación desarrollada para dispositivos móviles Android, se ha ejecutado en segundo plano la operación de enviar peticiones HTTP al servidor Apache con consultas a la base de datos MySQL y la operación de extracción de contenido de las páginas web a través de la librería JSoup.

- Extraer contenido HTML mediante la utilización de la librería JSoup ya que proporciona una API de gran utilidad para extraer y manipular datos. Dadas las características de la librería JSoup resulta ideal emplearla para realizar el *web scraping* y obtener los datos de interés sobre programación de televisión, películas y cartelera, requeridos por los módulos principales de la aplicación. La lógica empleada para realizar el *web scraping* no es compleja, únicamente se basa en seleccionar aquel código HTML que contiene la información que se quiere analizar.
- Carga asíncrona de imágenes remotas. Para realizar la carga asíncrona de imágenes remotas y su posterior almacenamiento en la memoria caché del dispositivo se ha partido de un código fuente sacado de un sitio web. El objetivo de dicho código es descargar las imágenes de la página web correspondiente y almacenarlas en la memoria caché del dispositivo, de esta manera, antes de cargar una imagen se comprueba si existe en la memoria local del dispositivo y así evitar su descarga. La carga asíncrona de imágenes remotas se utiliza en los módulos ***Búsqueda de películas*** y ***Cartelera***.

El balance obtenido del proyecto es, por lo tanto, muy positivo ya que se han alcanzado todos los objetivos parciales marcados inicialmente y gracias a ello, se ha logrado alcanzar el objetivo principal al desarrollar una aplicación multimodal que satisface los requisitos de los sistemas de diálogo ideales estudiados al inicio del proyecto y que se explican a continuación:

- La aplicación desarrollada reconoce sin dificultades el habla espontánea y comprende enunciados sin restricciones de contenido ya que integra el servicio de reconocimiento de voz ofrecido por Google, y Google, ha ido mejorando desde sus inicios dicho servicio hasta conseguir alcanzar estos requisitos.

- La aplicación se ha diseñado de forma que el sintetizador de voz proporcione respuestas con sentido, gramaticalmente bien formadas y pragmáticamente adecuadas. De hecho, el motor de síntesis de voz únicamente se encarga de reproducir las cadenas de texto que devuelve la aplicación, por lo que si se construyen de forma adecuada el presente requisito queda satisfecho.

- La aplicación responde con voz completamente natural ya que se ha utilizado el motor de síntesis IVONA TTS HQ cuya calidad es mejor que el resto de motores de síntesis estudiados. La voz ofrecida por IVONA TTS HQ es clara, natural y con buen acento.

- La aplicación ofrece una interacción multimodal a través del habla o de los interfaces tradicionales como el teclado o la pantalla.

Cabe destacar algunas de las consideraciones que se han tenido en cuenta a la hora de diseñar la aplicación multimodal para dispositivos móviles Android con el objetivo de mejorar la interacción oral entre el usuario y la aplicación. En primer lugar y con el objetivo de que la interacción oral entre el usuario y la aplicación sea lo más simple posible, se ha evitado utilizar locuciones demasiado largas, se han utilizado menús que no contengan un gran número de opciones y se ha diseñado la aplicación de forma que los niveles en la estructura de navegación sean los mínimos posibles. Además, la aplicación ofrece mecanismos de ayuda y sistemas de realimentación, que ayudan al usuario a saber que debe hacer y que está ocurriendo a medida que avanza por los diferentes menús de la aplicación. Por otro lado, para que el usuario se habitúe al modo de funcionamiento de la aplicación, se ha mantenido la coherencia y homogeneidad a lo largo de los diálogos.

La complejidad de la aplicación es elevada debido principalmente a la gran variedad de tecnologías que se han aplicado en su desarrollo, como Android, PHP, SQL, HTTP, JSON, JSoup, etc., muchas de ellas desconocidas anteriormente a la realización del presente proyecto. A continuación, se explican las principales dificultades que han sido las responsables de añadir complejidad a la realización de la aplicación para dispositivos móviles Android y se realizan algunas críticas.

- Antes de la realización del presente proyecto final de carrera se carecía de todo conocimiento acerca de la plataforma Android y del desarrollo de aplicaciones para la misma. Esto ha supuesto el tener que invertir un periodo de tiempo considerable al aprendizaje de dicha plataforma por lo que se ha visto retrasado el inicio del desarrollo de la aplicación.

- La extracción de contenido HTML de páginas web ha añadido complejidad al desarrollo de la aplicación ya que dichas páginas web han ido modificando su formato y contenido a lo largo de la realización del Proyecto Final de Carrera, por lo que ha sido necesario actualizar el código de acuerdo a los cambios en numerosas ocasiones. Especialmente, la página web *hoycinema.abc.es*, de la cual extrae información el módulo ***Cartelera***, cambió su formato totalmente lo que ocasionó que se tuviese que rehacer el código que se encargaba de extraer la información de la cartelera.

- La página web *Filmaffinity* restringió la descarga de imágenes, necesarias en el módulo ***Búsqueda de películas***. Como consecuencia, la aplicación descarga las imágenes de otra página web a pesar de no tener disponible imágenes para todas las películas como anteriormente.

- La interacción oral entre el usuario y la aplicación se ha visto dificultada al no existir un mecanismo en Android para que la síntesis de voz se detenga cuando el usuario desee comenzar a hablar. Esto resulta incómodo para el usuario considerando que este va aprendiendo cómo funciona la aplicación según va utilizándola, y no le resulta necesario esperar a que termine de hablar la aplicación para emitirle sus peticiones a través del reconocedor de voz.

- La aplicación no ofrece toda su funcionalidad si no dispone de acceso a Internet. Los módulos ***Cartelera*** y ***Búsqueda de películas*** no funcionan ya que extraen toda la información del contenido HTML de páginas web en el instante en el que se solicita. El módulo ***Asistente televisivo*** ofrece toda su funcionalidad, no obstante, si no se ha actualizado la base de datos con la programación del día actual, las recomendaciones televisivas estarán desactualizadas. Finalmente, no se puede utilizar el servicio de reconocimiento de voz cuando no se tiene acceso a internet, lo que imposibilita la interacción oral entre el usuario y aplicación.

- El reconocimiento de voz no funciona en modo *offline*, por lo que se necesita tener acceso a Internet para poder utilizar esta funcionalidad. Con la llegada de Android 4.1 *Jelly Bean*, se habilitó el reconocimiento de voz *offline* mediante la opción del menú Ajustes->Idioma y teclado->Búsqueda por voz->Reconocimiento de voz sin conexión, que permite escoger de forma individual los idiomas de los que se quiere hacer utilización del reconocimiento de voz *offline*. No obstante, no existe ninguna API disponible para implementar esta funcionalidad. Además, el reconocimiento de voz *offline* tiene como problema que no se encuentra disponible en todos los dispositivos y no existe ninguna documentación que haga referencia a su funcionalidad y a que dispositivos se puede aplicar. Por lo tanto, no se ha probado por el momento la aplicación con el reconocimiento de voz *offline* ya que el dispositivo con el que se hicieron las pruebas no disponía de esta funcionalidad.

- La aplicación no implementa un mecanismo de siempre a la escucha, capaz de quedarse esperando a que el usuario hable para activarse, lo que resultaría de gran interés para entornos en los que es imposible el uso de interfaces tradicionales (por ejemplo, en conducción).

- Cuando el sintetizador está leyendo un texto en un determinado idioma, no diferencia las cadenas dentro del texto que están en otro idioma por lo que las pronuncia en el idioma en el que se estaba efectuando la síntesis de texto a voz. Esto dificulta el entendimiento de los mensajes generados por el sistema. Esto se podría solucionar ya que el diseñador puede especificar la pronunciación de determinadas cadenas de texto. Sin embargo, es muy difícil tener en cuenta todas las posibles cadenas existentes y el texto podría escucharse entrecortado.

A pesar de las dificultades y críticas expuestas, el balance del proyecto es muy positivo al haberse cumplido todos los objetivos marcados inicialmente conllevando al desarrollo de una aplicación multimodal cuya ventaja principal es la de ser capaz de facilitar el acceso a los servicios de cine y de televisión a personas con problemas de visión o discapacidades motoras, o en entornos en los que no es aconsejable o resulta imposible el uso de interfaces tradicionales (por ejemplo, en conducción). De acuerdo a los resultados obtenidos en el apartado de evaluación, el modo multimodal ha sido el mejor valorado por los usuarios ya que el usuario prefiere combinar el modo táctil y el modo oral según le resulte más eficaz, aprovechando todas las ventajas y evitando todas las desventajas de ambos modos.

6.2. Trabajo futuro

Como líneas futuras de trabajo, se proponen los siguientes puntos:

- Mejoras en el módulo ***Registro e inicio de sesión***: Dotar de mayor seguridad al sistema a través de un algoritmo de encriptación de contraseñas que ayude a proteger las contraseñas ante posibles vulnerabilidades en el servidor, de forma que si alguien puede acceder a ellas no pueda ver la contraseña si no su encriptación.
- Mejoras en el módulo ***Búsqueda de películas***: Este módulo únicamente funciona si la aplicación tiene acceso a Internet ya que realiza consultas a la página web *Filmaffinity* para obtener información acerca de una película en el instante que la necesita. Las mejoras que se proponen a continuación tienen como objetivo hacer más independiente a la aplicación del estado de la conexión.
    - Para evitar que este servicio deje de funcionar en su totalidad en el caso de que se caiga el servidor de *Filmaffinity*, se propone implementar una operación que almacene, en una tabla de la base de datos MySQL del servidor web remoto, toda la información acerca de las películas que el usuario vaya consultando. De esta forma, cuando el usuario solicite información sobre una película, primero se realizará la consulta en la base de datos MySQL del servidor web, y únicamente en el caso de que no exista una entrada en dicha base de datos coincidente con la película solicitada, se extraerá la información de la página web *Filmaffinity* y se guardará en la base de datos MySQL del servidor web para futuras consultas. De esta forma, a medida que los usuarios utilicen este servicio, la tabla de la base de datos MySQL contendrá información sobre un mayor número de películas.
    - Con el objetivo de que el usuario pueda consultar información acerca de una determinada película sin tener acceso a Internet, se propone implementar una operación que almacene, en una tabla de la base de datos SQLite del dispositivo, toda la información acerca de las películas que el usuario vaya consultando. De esta forma, cuando el usuario solicite información sobre una película, primero se realizará la consulta en la base de datos SQLite del dispositivo, y únicamente en el caso de que no exista una entrada en dicha base de datos coincidente con la película solicitada, se extraerá la información de la página web *Filmaffinity* y se guardará en la base de datos SQLite del dispositivo para futuras consultas. De esta forma, a medida que el usuario utilice este servicio, la tabla de la base de datos SQLite contendrá información sobre un mayor número de películas. Con el objetivo de no alcanzar el máximo de almacenamiento de la base de datos SQLite, se fijará un máximo de entradas que la tabla de base de datos SQLite puede contener y en el caso de alcanzar este máximo se vaciará la tabla. El algoritmo es similar al utilizado en la carga asíncrona de imágenes remotas, en el que se van almacenando las imágenes en la caché del dispositivo.
- Mejoras en el módulo ***Recomendación de películas***:
    - Este módulo accede a una tabla de la base de datos SQLite que contiene una lista de 100 películas basada en los gustos de cine del usuario. A medida que el módulo devuelve las recomendaciones al usuario, este puede solicitar información más específica acerca de la película, documental o serie de televisión recomendada ya que el módulo ***Recomendación***

de películas dirige al usuario al módulo ***Búsqueda de películas*** en el caso de que este lo requiera. Para que la aplicación no necesite tener acceso a Internet para solicitar información acerca de estas 100 películas, se propone ampliar los campos de la tabla que contiene la lista de las 100 películas de forma que además almacene toda la información de las películas (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios). Dado que la operación de consultar dicha información a la página web *Filmaffinity* puede requerir un intervalo de tiempo considerable, es conveniente que se implemente esta operación en el lado del servidor por lo que la base de datos MySQL del servidor web remoto también deberá contener esta tabla. La información de la tabla con la lista de las 100 películas del servidor web remoto se enviará a la tabla correspondiente de la base de datos SQLite cuando el usuario se identifique. De esta forma, cuando el usuario solicite información más específica acerca de la película, documental o serie de televisión recomendada, la aplicación no necesitará tener acceso a Internet ya que toda la información necesaria estará almacenada en la base de datos SQLite interna.

- Implementar un algoritmo en el servidor que se encargue de calcular el porcentaje de afinidad entre usuarios. Además, los usuarios podrán evaluar las películas recomendadas una vez vistas y dicha evaluación se almacenará en la tabla de la base de datos SQLite y posteriormente se enviará a la tabla de la base de datos MySQL del servidor web remoto. El servidor se encargará de implementar un algoritmo que recomiende las películas mejor valoradas entre usuarios con un porcentaje de afinidad elevado.

- Mejoras en el módulo ***Cartelera***: Este módulo necesita tener acceso a Internet para ofrecer toda su funcionalidad al usuario por lo que se proponen dos mejoras al respecto.

  - Permitir al usuario seleccionar sus cines favoritos y que la base de datos SQLite del dispositivo contenga una tabla con la información de cartelera de dichos cines. La información de dicha tabla se podrá actualizar siempre que el usuario lo requiera, de la misma forma que se actualiza la tabla con la programación televisiva diaria en el módulo ***Asistente televisivo***. Es recomendable que el usuario actualice la información de dicha tabla cada día.
  - Disponer de una tabla que contenga toda la información (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios) acerca de las películas en cartelera del cine favorito del usuario en la base de datos SQLite del dispositivo. Dado que la operación de consultar dicha información a la página web *Filmaffinity* puede requerir un intervalo de tiempo considerable, es conveniente que se implemente esta operación en el lado del servidor por lo que la base de datos MySQL del servidor web remoto deberá contener también esta tabla. La información de esta tabla del servidor web remoto se enviará a la tabla correspondiente de la base de datos SQLite cuando el usuario se identifique. De esta forma, cuando el usuario solicite información más específica acerca de la película en cartelera, la aplicación no necesitará tener acceso a Internet ya que toda la información necesaria estará almacenada en la base de datos SQLite interna.

  Además, se sugiere también como mejora el implementar un algoritmo de localización de cines más cercanos. Para ello, se debe utilizar el componente GPS presente en el mismo dispositivo móvil para conocer la ubicación actual del usuario.

- Mejoras en el módulo ***Asistente televisivo***:

  - Ampliar el cuestionario inicial para que la aplicación sea más restrictiva a la hora de recomendar la programación televisiva al usuario. Por ejemplo: filtrar resultados en cuanto a tipo de programa televisivo (salsa rosa, informativos, debates, entrevistas, etc.), filtrar resultados en cuanto a género de películas, etc.
  - Ir mejorando la inteligencia del algoritmo del asistente de forma continua añadiendo palabras claves asociadas cada categoría televisiva para la búsqueda de coincidencias.
  - Añadir un mayor número de canales televisivos a la base de datos.

  - Disponer de una tabla en la base de datos SQLite del dispositivo que contenga toda la información (sinopsis, título original, país, duración, director, reparto, género, año, calificación y premios) acerca de las películas, series y documentales que se emiten diariamente en televisión únicamente si el usuario ha registrado en el cuestionario inicial que desea escuchar recomendaciones televisivas acerca de cine. Dado que la operación de consultar dicha información a la página web *Filmaffinity* puede requerir un intervalo de tiempo considerable, es conveniente que se implemente esta operación en el lado del servidor por lo que la base de datos MySQL del servidor web remoto deberá contener también esta tabla. La información de esta tabla del servidor web remoto se enviará a la tabla correspondiente de la base de datos SQLite cuando el usuario se identifique y únicamente si el usuario ha registrado en el cuestionario inicial que desea escuchar recomendaciones televisivas acerca de cine. De esta forma, cuando el usuario solicite información más específica acerca de la película, serie o documental que emiten en la televisión, la aplicación no necesitará tener acceso a Internet ya que toda la información necesaria estará almacenada en la base de datos SQLite interna.

- Mejoras en el módulo ***Asistente por voz***:

  - Implementar un mecanismo de "Siempre a la escucha" de forma que la aplicación se quede a la espera de que el usuario diga una palabra clave para comenzar el diálogo.
  - A pesar de no ser posible actualmente, se deberá conseguir en un futuro que la síntesis de texto a voz se detenga siempre que el usuario hable. De esta forma, no será necesario que el usuario escuche todas las instrucciones si sabe lo que va contestar de antemano, de modo que se consiga un acceso más rápido y natural.
  - Incrementar la realimentación. Por ejemplo: preguntar al usuario si el reconocedor ha entendido bien la película sobre la que el usuario solicita información, preguntar al usuario si el reconocedor ha entendido bien la provincia en la que se desea encontrar las salas de cine, etc.

  .

- Permitir que funcione el reconocimiento de voz *offline,* lo que permite utilizar el reconocimiento de voz sin la necesidad de tener acceso a Internet. Por el momento, no existe ninguna API disponible para implementar esta funcionalidad y se activa desde la opción del menú Ajustes->Idioma y teclado->Búsqueda por voz->Reconocimiento de voz sin conexión, que permite escoger de forma individual los idiomas de los que se quiere hacer utilización del reconocimiento de voz *offline*. El problema de esta funcionalidad es que no se encuentra disponible en todos los dispositivos y no existe ninguna documentación que haga referencia a esta funcionalidad y a que dispositivos se puede aplicar por lo que no se ha incorporado por el momento en la aplicación desarrollada.

- Añadir nuevos módulos a la aplicación: ***Próximos estrenos***, ***Premios de cine***, ***Noticias de cine y televisión***, etc.

Capítulo 7

GESTIÓN DEL PROYECTO

7.1. Planificación temporal

En base a la división del proyecto en fases y tareas descrita en la Sección 1.3, se ha realizado la planificación temporal utilizando la herramienta *GanttProject*, que permite crear un diagrama de Gantt. El diagrama de Gantt tiene como objetivo mostrar el tiempo que se ha dedicado a las fases y tareas en las que ha sido dividido el proyecto.

La Figura 7.1 muestra una tabla realizada con la herramienta *GanttProject* que contiene las fechas y duración estimadas de cada tarea. La tabla incluye las 3 fases principales del proyecto (Planificación, Ejecución y Documentación) que agrupan a las 11 tareas estudiadas en el diagrama WBS de la Sección 1.3. La duración de cada tarea se establece considerando una dedicación de 3 horas al día aproximadamente de lunes a jueves, debido a la compaginación con el trabajo. Se ha excluido del calendario del proyecto los viernes, sábados y domingos al no dedicar tiempo estos días en general al desarrollo del proyecto.

Nombre	Fecha de inicio	Fecha de fin	Duración
Inicio del proyecto	2/10/12	2/10/12	0
Fase de planificación	2/10/12	27/06/13	155
Estudio de los sistemas d...	2/10/12	25/10/12	15
Estudio de la plataforma A...	29/10/12	18/12/12	30
Análisis de alternativas pa...	19/12/12	26/02/13	40
Estudio de aplicaciones si...	27/02/13	14/03/13	10
Definición de los requisito...	18/03/13	2/04/13	10
Estudio de las tecnologías...	3/04/13	27/06/13	50
Fase de Ejecución	5/08/13	19/05/14	165
Diseño detallado	5/08/13	28/08/13	15
Programación de la aplica...	29/08/13	25/11/13	50
Integración y pruebas	26/11/13	3/02/14	40
Evaluación de la aplicación	1/05/14	19/05/14	10
Fase de cierre	5/08/13	23/06/14	185
Redacción de la memoria	5/08/13	27/05/14	170
Preparación de la present...	28/05/14	23/06/14	15
Fin de proyecto	24/06/14	24/06/14	0

Figura 7.1: Estimación de las fechas y duración empleadas en cada tarea

La Figura 7.2 muestra el diagrama de Gantt resultante de aplicar la planificación temporal de la

tabla de la Figura 7.1. Tal y como se observa, este diagrama vincula las tareas de dicha tabla a un calendario permitiendo de esta forma un seguimiento detallado del estado de avance de cada tarea. Además, se ha añadido un hito de inicio de proyecto y un hito de fin de proyecto. Un hito es un evento de duración nula que controla el inicio o finalización de un grupo de tareas en un proyecto y tiene como objetivo mejorar el control del proyecto. Por último, se observa las relaciones de precedencia de cada tarea, que fijan la restricción de no empezar una cierta tarea si su tarea precedente no ha finalizado, exceptuando la tarea de redacción de la memoria, que se solapa en el tiempo con la fase de ejecución ya que al igual que esta, se inicia una vez terminada la fase de planificación.

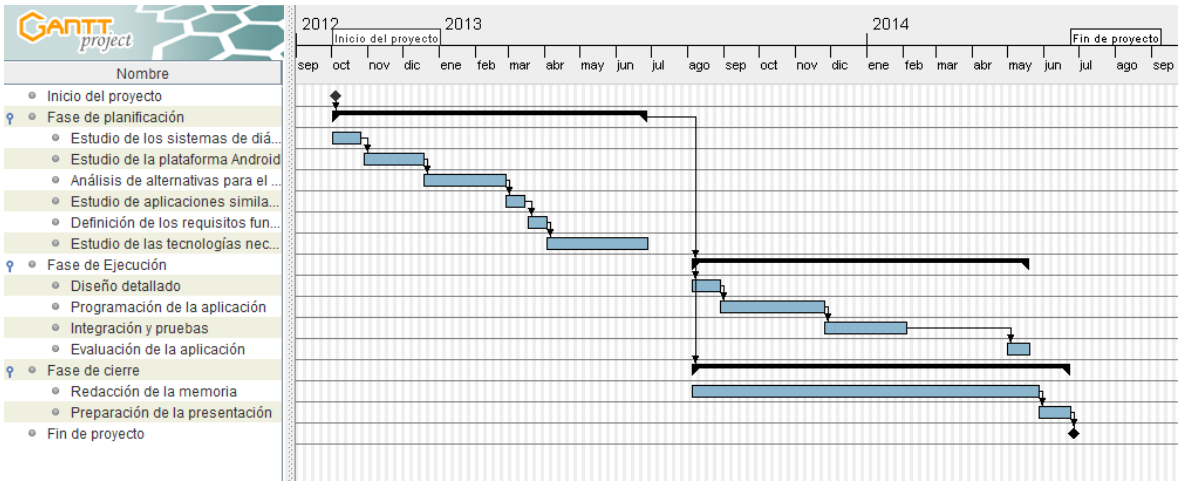


Figura 7.2: Diagrama de Gantt de la planificación temporal del Proyecto Final de Carrera

Por lo tanto, se ha estimado una duración de 340 días para la realización del proyecto, sin incluir los días en los que no se ha ejecutado ninguna tarea o bien por tratarse de un viernes, sábado o domingo o bien debido a los retrasos entre finalización de una tarea y comienzo de la siguiente que se observa en la Figura 7.2. Teniendo en cuenta que la fase de Documentación se realizó simultaneamente a la fase de Ejecución y que ambas fases se inician en la misma fecha y que la fase de Documentación termina después, se ha cogido la duración de la fase de Documentación en lugar de la duración de la fase de Ejecución para el cálculo de la duración total del proyecto.

El tiempo invertido en la tarea de planificación ha sido mayor al esperado debido a la labor de investigación y estudio de la plataforma Android ya que se desconocía anteriormente y al estudio de las tecnologías necesarias para el desarrollo de la aplicación, por lo que la finalización del proyecto ha sufrido un retraso considerable.

7.2. Presupuesto

En base a la planificación y duración del proyecto que se han establecido en el apartado anterior, se incluye en este apartado el desglose presupuestario del proyecto que incluye los costes directos (costes de personal y costes de equipo) y costes indirectos (20 %). Como modelo de ayuda para confeccionar el presupuesto del presente proyecto, se utiliza la plantilla que proporciona la universidad (uc3m, 2014).

Costes de Personal

A partir de la formula:

Coste = ((duración días * horas diarias)/dedicación hombre mes) * coste hombre mes

Se calcula el coste de personal, teniendo en cuenta que para el presente proyecto:

- Duración días = 340
- Horas diarias = 3
- Dedicación hombre mes = 131,25
- Coste hombre mes = 2694,39 (ingeniero)

Dando como resultado unos gastos de personal de 20.939,26 €.

Costes de Equipo

Para la elaboración del proyecto ha sido necesario adquirir:

- Recursos Hardware
  - Ordenador portátil: 900 €.
  - Smartphone HTC Desire X: 200 €.
  - Cable usb: 5 €.
  - Servidor web x10Hosting: 0 €.
- Recursos Software
  - Entorno de desarrollo integrado de código abierto multiplataforma Eclipse: 0 €.
  - SDK (*Software Development Kit*) de Android: 0 €.
  - JDK (*Java development kit*): 0 €.
  - Plug-in ADT (Android Development Tools) para Eclipse: 0 €.
  - phpMyAdmin: 0 €.
  - JSoup: 0 €.
  - Sintetizador de voz IVONA TTS: 0 €.
  - Aplicación de búsqueda por voz de Google: 0 €.
  - LyX: 0 €.
  - Herramienta *GanttProject* para la creación del diagrama de Gantt: 0 €.
  - Servicio de alojamiento de archivos multiplataforma *Dropbox*: 0 €.

La Tabla 7.1 muestra la amortización de Equipos, calculada siguiendo la plantilla.

Descripción	Coste (€)	% Uso dedicado proyecto	Dedicación (meses)	Periodo de depreciación	Coste imputable
Ordenador portátil	900	100	7,86	60	117,90
Smartphone HTC Desire X	200	100	7,86	60	26,20
Cable usb	5	100	7,86	60	0,66
				Total	144,76

Tabla 7.1: Amortización de Equipos

La fórmula de cálculo de la Amortización utilizada es:

$$\frac{A}{B} \text{xCxD}$$

Donde,

- A = número de meses desde la fecha de facturación en que el equipo es utilizado
- B = periodo de depreciación (60 meses)
- C = coste del equipo (sin IVA)
- D = % del uso que se dedica al proyecto (habitualmente 100 %)

Se ha tenido en cuenta para calcular la dedicación, que únicamente se han utilizado los equipos durante la realización de las siguientes tareas: estudio de la plataforma Android, análisis de alternativas para el desarrollo de aplicaciones para dispositivos móviles Android basadas en sistemas de diálogo, estudio de aplicaciones para dispositivos móviles Android basadas en sistemas de diálogo, estudio de las tecnologías necesarias para desarrollar la aplicación, programación de la aplicación, e integración y Pruebas.

Finalmente, la Tabla 7.2 muestra el resumen de costes del presente proyecto.

Presupuesto Costes Totales	Presupuesto Costes Totales (€)
Personal	20.939,26
Amortización	144,76
Subcontratación de tareas	0
Costes de funcionamiento	0
Costes Indirectos (20 %)	4.216,80
Total sin IVA	25.300,82
Total con IVA (21 %)	30.613,99

Tabla 7.2: Resumen de Costes

El presupuesto total de este proyecto asciende a la cantidad de TREINTA MIL SEISCIENTOS CATORCE EUROS.

Madrid a 2 de Mayo de 2014

Fdo. Irene Marquet Gómez

GLOSARIO

- **Activity**: componente de una aplicación Android que refleja una actividad llevada a cabo por una aplicación, y que lleva asociada típicamente una interfaz de usuario (vistas representadas por subclases View). Este componente se implementa mediante la clase Activity.
- **API (Application Programming Interface)**: conjunto de llamadas que ofrecen acceso a funciones y procedimientos, representando una capa de abstracción para el desarrollador.
- **ASR (Automatic Speech Recognition)**: módulo de un sistema de diálogo que se encarga de reconocer la señal de voz emitida por el usuario y devolver la(s) secuencia(s) de palabras reconocida(s) más probable mediante la aplicación de técnicas de proceso de señal de voz. Las siglas en español son RAH (Reconocimiento Automático del habla).
- **Asynctask**: clase que permite ejecutar operaciones en segundo plano e ir publicando los resultados en un hilo de la interfaz gráfica. De esta forma, utilizando programación concurrente es posible realizar tareas de forma concurrente sin realizar operaciones bloqueantes.
- **APK (Application PacKage File)**: paquete para el sistema operativo Android.
- **Background**: representa un proceso que se ejecuta en segundo plano.
- **Bluetooth**: protocolo que permite la transmisión de datos entre dispositivos, más o menos próximos y alineados, mediante un enlace de radiofrecuencia. Está especialmente diseñado para dispositivos de bajo consumo y coste.
- **Broadcast Intent Receiver**: componente de una aplicación Android utilizado para lanzar alguna ejecución dentro de la aplicación actual cuando un determinado evento se produce (generalmente, abrir un componente Activity). Este componente se implementa a través de la clase BroadcastReceiver.
- **Bytecode**: código intermedio, más abstracto que el código máquina, y que necesita de un mediador o máquina virtual para poder ser transformado y ejecutado en un hardware local.
- **CDMA (Code Division Multiple Access)**: métodos de multiplexación o control de acceso al medio basados en la tecnología de espectro expandido.
- **CGI (Common Gateway Interface)**: es una importante tecnología de la World Wide Web que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor web y una aplicación externa cuyo resultado final de la ejecución son objetos MIME. Las aplicaciones que se ejecutan en el servidor reciben el nombre de CGIs.
- **Content Providers**: gestor de contenidos que permite a cualquier aplicación en Android almacenar datos en un fichero, en una base de datos SQLite o en cualquier otro formato. Una clase que implemente dicho componente contendrá una serie de métodos que permiten almacenar, recuperar, actualizar y compartir datos entre distintas aplicaciones. Existe una colección de clases para distintos tipos de gestión de datos en el paquete android.provider.

- **Controlador**: programa que permite al sistema operativo interactuar con un periférico, abstrayéndolo y proporcionando una interfaz para usarlo. También conocido como driver.
- **CSS (Cascading Style Sheets)**: lenguaje de hojas de estilo utilizado para describir el aspecto y formato de un documento escrito en un lenguaje de marcas.
- **Dalvik**: nombre de la máquina virtual utilizada por el sistema operativo Android. Dalvik esta específicamente adaptada a las características de rendimiento de un dispositivo móvil y trabaja con ficheros de extensión ".dex", obtenidos desde el bytecode de Java.
- **DOM (Document Object Model)**: API que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.
- **EDGE (Enhanced Data rates for GSM of Evolution)**: tecnología para telefonía móvil que representa un puente entre la segunda y tercera generación de estos dispositivos.
- **Elocuciones**: En inglés *utterances*, son las cadenas de texto que esperan a ser sintetizadas en términos TTS.
- **E/S**: abreviatura de Entrada/Salida. Un elemento de E/S es aquél que permite la comunicación entre un sistema de procesamiento de datos y una entidad externa a él (un usuario humano u otro sistema de procesamiento). Entrada se considera toda aquella información que es recibida por un sistema, mientras que salida es aquella información que es enviada por el mismo.
- **FTP (File Transfer Protocol)**: protocolo de red para la transferencia de ficheros entre sistemas conectados a una red TCP (*Transmission Control Protocol*) basado en la arquitectura cliente-servidor.
- **GPL (General Public License)**: licencia que permite utilizar el software MySQL de forma gratuita.
- **GPS (Global Positioning System)**: sistema global de navegación que, mediante satélites, permite ubicar un objeto en la superficie terrestre con una precisión que va desde varios metros a centímetros.
- **GSM (Global System for Mobile communications)**: estándar para las comunicaciones de telefonía móvil digital. Permite llamadas, navegación por Internet o envío de SMS.
- **HTML (HyperText Markup Languaje)**: lenguaje de marcado para la elaboración de páginas web.
- **HTTP (Hypertext Transfer Protocol)**: desarrollado por W3C e IETF, es un protocolo de nivel de aplicación de la web que sigue el esquema petición-respuesta entre un cliente y un servidor.
- **IDE (Integrated Development environment) de Eclipse**: entorno de desarrollo integrado de código abierto multiplataforma.
- **IETF (Internet Engineering Task Force)**: organización internacional abierta de normalización, que tiene como objetivos el contribuir a la ingeniería de Internet, actuando en diversas áreas, como transporte, encaminamiento y seguridad.
- **IME (Input Method)**: método de entrada a un sistema.
- **Intent**: componente de una aplicacion Android que consiste en la voluntad de realizar alguna acción, generalmente asociada a unos datos. Lanzando un *intent*, una aplicación puede delegar el trabajo en otra (Ej. abrir una URL en algún navegador web), de forma que el sistema se encarga de buscar qué aplicación entre las instaladas es la que puede llevar a cabo la acción solicitada. Este componente se implementa a través de la clase `Intent`.

- **JDK (Java Development Kit)**: kit de desarrollo de Java.
- **jQuery**: biblioteca de JavaScript, que simplifica la forma de actuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.
- **JSON (JavaScript Object Notation)**: formato ligero de intercambio de datos. El formato de texto JSON es completamente independiente del lenguaje con el que se está programando pero utiliza convenciones ampliamente utilizadas en distintos lenguajes de programación. Por lo tanto, JSON en un lenguaje ideal para el intercambio de datos.
- **JSoup**: librería Java que permite al usuario trabajar con HTML. Proporciona una API de gran utilidad para extraer y manipular datos, utilizando métodos DOM, CSS y jQuery.
- **Middleware**: capa de abstracción software que posibilita el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas.
- **Motor de síntesis**: motor que permite integrar la síntesis de texto a voz a cualquier aplicación. Mediante el motor de síntesis de voz, las aplicaciones son capaces de producir voz a partir de cualquier cadena de texto, permitiendo al usuario interactuar con la aplicación sin la necesidad de leer la pantalla del dispositivo. Todo dispositivo Android incorpora por lo general un motor TTS instalado por defecto, mediante el cual las aplicaciones son capaces de leer textos en voz alta. Sin embargo, Android ofrece la posibilidad de instalar y personalizar varios motores, aunque siempre eligiendo uno como motor TTS principal.
- **MySQL**: MySQL es un sistema de gestión de bases de datos relacionales desarrollado y distribuido por la compañía comercial MySQL AB. El software de base de datos MySQL proporciona un servidor de base de datos SQL muy rápido, multihilo, multiusuario y robusto. El software MySQL tiene una doble licencia. Los usuarios pueden elegir entre usar el software MySQL de forma gratuita bajo licencia GNU GPL (*General Public License*) pueden adquirir una licencia comercial estándar de MySQL AB.
- **OHA (Open Handset Alliance)**: conglomerado de empresas de sectores tecnológicos lideradas por Google que promueve la innovación y desarrollo de dispositivos móviles y sus aplicaciones. Su primera contribución es el sistema operativo Android.
- **PDO (PHP Data Objects)**: define una interfaz ligera para poder acceder a bases de datos en PHP. No se puede realizar ninguna de las funciones de la base de datos utilizando la extensión PDO por sí misma sino que se debe utilizar un controlador de PDO específico de la base de datos para tener acceso a un servidor de bases de datos.
- **PHP (Hypertext Preprocessor)**: lenguaje de código abierto utilizado en el desarrollo web y que puede ser incrustado en HTML. Principalmente, PHP está enfocado a la programación de scripts en el lado del servidor.
- **PhpMyAdmin**: administrador de bases de datos MySQL que se utiliza vía web.
- **plug-in: componente de software que se relaciona y ejecuta con otro para aportarle una función nueva y generalmente muy especifica.**
- **plug-in ADT (Android Development Tools)**: plug-in para Eclipse que permite desarrollar aplicaciones Android utilizando Eclipse
- **RAH (Reconocimiento Automático del habla)**: módulo de un sistema de diálogo que se encarga de reconocer la señal de voz emitida por el usuario y devolver la(s) secuencia(s) de palabras reconocida(s) más probable mediante la aplicación de técnicas de proceso de señal de voz. Las siglas en inglés son ASR (*Automatic Speech Recognition*).
- **RecognizerIntent**: clase del paquete android.speech que define las constantes necesarias para integrar el reconocimiento de voz iniciado desde un *intent*.

- **SDK (Software Development Kit)**: constituye un conjunto de herramientas que permiten a un desarrollador crear aplicaciones para una determinada plataforma o lenguaje.
- **Sendmail**: es un popular agente de transporte de correo (MTA - *Mail Transport Agent*) en Internet, cuya tarea consiste en encaminar los mensajes de forma que estos lleguen a su destino. Se afirma que es el más popular MTA, compatible con sistemas Unix y el responsable de la mayoría de envíos del correo de Internet, aunque se critica su alto número de alertas de seguridad además de no ser sencillo de configurar.
- **Service**: componente de una aplicación para Android que representa una aplicación ejecutada sin interfaz de usuario, generalmente en segundo plano, mientras otras aplicaciones (con interfaz) están activas en la pantalla del dispositivo. Este componente se implementa a través de la clase Service.
- **Sistema de diálogo**: programas informáticos cuya finalidad es interactuar con los usuarios oralmente o de forma multimodal para proporcionar diversos servicios.
- **Smartphone**: dispositivo móvil que representa una evolución de los teléfonos móviles, con la inclusión de pantalla táctil, teclado, conexión Wi-Fi, aplicaciones de usuario como navegador web o cliente de correo, entre otros.
- **SMTP (Simple Mail Transfer Protocol)**: protocolo para la transferencia simple de correo electrónico.
- **SQL (Structured Query Language)**: lenguaje de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.
- **SQLite**: motor de bases de datos SQL que ocupa poco tamaño, no necesita servidor, precisa poca configuración, es transaccional y es de código libre. Android incorpora todas las herramientas necesarias para la creación y gestión de bases de datos SQLite, y entre ellas una API para llevar a cabo de manera sencilla todas las tareas necesarias.
- **TextToSpeech**: clase del paquete android.speech.tts que se encarga de la síntesis de texto a voz.
- **TTS (Text-to-Speech)**: módulo de un sistema de diálogo que se encarga de transformar la respuesta que recibe del sistema como texto en lenguaje natural, en la señal de audio correspondiente que recibirá el usuario.
- **Web Scraping**: técnica utilizada para extraer información de sitios web.
- **Widget**: componente gráfico utilizado en interfaces de usuario, con el cual el usuario puede interactuar, como por ejemplo cajas de texto, botones, ventanas, etc.
- **Wi-Fi (Wíreless Fidelity)**: estándar de envío de datos que utiliza ondas de radio en lugar de cables.
- **W3C (World Wide Web Consortium)**: asociación internacional formada por organizaciones miembros del consorcio, personal y el público en general, que trabajan conjuntamente para desarrollar estándares web. W3C pretende guiar la web hacia su máximo potencial a través del desarrollo de protocolos y pautas que aseguren el crecimiento futuro de la web.
- **XML (Extensible Markup Language)**: representa un lenguaje estándar que, mediante el uso de etiquetas y atributos, permite expresar e intercambiar fácilmente estructuras de datos.
- **x10hosting**: un servidor web que incluye bases de datos MySQL, un servidor web Apache y el intérprete para lenguaje de script PHP 5.
- **3G**: abreviación de tercera generación de transmisión de voz y datos a través de telefonía móvil mediante UMTS (*Universal Mobile Telecommunications System* o servicio universal de telecomunicaciones móviles).

REFERENCIAS

The 7 Best Android Text-To-Speech Engines. (2012). Descargado Agosto, 2013, de http://www.geoffsimons.com/2012/06/7-best-android-text-to-speech-engines.html#!/2012/06/7-best-android-text-to-speech-engines.html

Achour, M., Betz, F., Dovgal, A., Lopes, N., Magnusson, H., Richter, G., y Seguy, D. (2013). *Manual de PHP.* Descargado Agosto, 2013, de http://www.php.net/manual/es/

Android APIs. (2014). Descargado Febrero, 2014, de http://developer.android.com/reference/packages.html

Apache HttpComponents. (2014). Descargado Mayo, 2014, de http://hc.apache.org/

App gallery. (2014). Descargado Agosto, 2013, de http://svoxmobilevoices.wordpress.com/

Apple Computer, Inc., Mozilla Foundation, y Opera Software ASA. (2014). *HTML: The living standard.* Descargado Mayo, 2014, de http://www.whatwg.org/specs/web-apps/current-work/multipage/

Aranaz, J. (2009). *Desarrollo de aplicaciones para dispositivos móviles sobre la plataforma android de google* (Proyecto Fin de Carrera). Madrid, España: Universidad Carlos III.

Barra, H. (2010). Just speak it: introducing Voice Actions for Android. En *Google Official Blog.* Descargado Mayo, 2014, de http://googleblog.blogspot.com.es/2010/08/just-speak-it-introducing-voice-actions.html

Benyon, D., y Mival, O. (2007). Introducing the COMPANIONS project: Intelligent, persistent, personalised multimodal interfaces to the internet. En C. Sas y T. Ormerod (Eds.), *21st british HCI group annual conference* (Vol. 2, pp. 193–194). Swinton, UK. doi: 978-1-902505-95-4

Cabero, G., y Maldonado, D. (2007). *Sqlite: Rápido, ágil, liviano y robusto.* Descargado Mayo, 2014, de http://es.scribd.com/doc/124083358/52882068-SQLite

Cardinal, D. (2012). *Google's knowledge graph: Wikipedia on steroids, or the beginning of the end for the web?* Descargado Mayo, 2014, de http://www.extremetech.com/computing/129566-googles-knowledge-graph-wikipedia-on-steroids-or-the-beginning-of-the-end-for-the-web

CMU Sphinx. (2013). Descargado Mayo, 2014, de http://cmusphinx.sourceforge.net/wiki/

Cómo usar tu voz en Android. (2014). Descargado Mayo, 2014, de https://support.google.com/websearch/answer/2940021

Corrales, V. (2010). *Desarrollo de un entorno para la interacción multimodal con diferentes aplicaciones en XHTML+voice* (Proyecto Fin de Carrera). Madrid, España: Universidad Carlos III.

Criado, V. (2012). *Análisis completo del S voice del galaxy S3 (Parte I).* Descargado Agosto, 2013, de http://androidayuda.com/2012/05/24/analisis-completo-del-s-voice-del-galaxy-s3-parte-i/

eGuidedog. (2014). Descargado Mayo, 2014, de http://www.eguidedog.net/

eSpeak text to speech. (2014). Descargado Mayo, 2014, de http://espeak.sourceforge.net/

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., y Berners-Lee, T. (1999). *Hypertext Transfer Protocol – HTTP 1.1.* Descargado Mayo, 2014, de http://tools.ietf.org/html/rfc2616

Flite: a small, fast run time synthesis engine. (2014). Descargado Mayo, 2014, de http://www.speech.cs.cmu.edu/flite/

Flite TTS Engine for Android. (2014). Descargado Mayo, 2014, de https://github.com/happyalu/Flite-TTS-Engine-for-Android

García, M. (2011). *Desarrollo de un Portal de Voz de Atención al Ciudadano mediante VoiceXML* (Proyecto Fin de Carrera). Madrid, España: Universidad Carlos III.

Google España. (2011). *Acciones de voz* [Archivo de video]. Descargado Mayo, 2014, de http://www.youtube.com/watch?feature=player_embedded&v=o6L8DnUNPV0

Google I/O. (2013). Descargado Mayo, 2014, de https://developers.google.com/events/io/

Google Mobile. (2010). *Introducing voice Actions for Android* [Archivo de video]. Descargado Mayo, 2014, de http://www.youtube.com/watch?v=gGbYVvU0Z5s

Google Nexus One. (2010). *Nexus One-Voice Input.*

Google Play. (2014). Descargado Mayo, 2014, de https://play.google.com/

Griol, D. (2007). *Desarrollo y Evaluación de Diferentes Metodologías para la Gestión Automática del Diálogo* (Tesis Doctoral). Valencia, España: Universidad Politécnica de Valencia.

Griol, D., Callejas, Z., López-Cózar, R., y Gutiérrez, A. (2009, octubre). *Utilización de los sistemas de diálogo hablado para el acceso a la información en diferentes dominios.* II Conferencia Internacional sobre Brecha Digital e Inclusión Social, Leganés, España.

Griol, D., Hurtado, L., Segarra, E., y Sanchis, E. (2008). A statistical approach to spoken dialog systems design and evaluation. En *Speech Communication* (Vol. 50, pp. 666–682). doi: 10.1016/j.specom.2008.04.001

Gruenstein, A. (2010). *Speech Input API for Android.* Descargado Mayo, 2014, de http://android-developers.blogspot.com.es/2010/03/speech-input-api-for-android.html

Hashimi, S., Komatineni, S., y MacLean, D. (2010). Exploring Text to Speech and Translate APIs. En S. Hashimi, S. Komatineni, y D. MacLean (Eds.), *Pro Android 2* (pp. 583–589). United States of America: Apress.

Hernández, J. (2012). *10 Sintetizadores de voz (TTS) para Android.* Descargado Mayo, 2014, de http://www.emezeta.com/articulos/10-sintetizadores-de-voz-tts-para-android#axzz2GjM77pDz

HMM-based Speech Synthesis System (HTS). (2013). Descargado Mayo, 2014, de http://hts.sp.nitech.ac.jp/

Holly, R. (2012). *Android 4.1 (Jelly Bean) Voice Actions explained.* Descargado Mayo, 2014, de http://www.geek.com/mobile/googles-new-jelly-bean-voice-actions-1499437/

Introducing JSON. (2014). Descargado Mayo, 2014, de http://www.json.org/

Jiménez, E. (2012). *S Voice, el Siri de Samsung que llega al Galaxy S3.* Descargado Mayo, 2014, de http://androidayuda.com/2012/05/04/s-voice-el-siri-de-samsung-que-llega-al-galaxy-s3/

JSoup. (2013). Descargado Mayo, 2014, de http://androidayuda.com/2012/05/04/s-voice-el-siri-de-samsung-que-llega-al-galaxy-s3/

Jupiter. (2013). Descargado Mayo, 2014, de http://groups.csail.mit.edu/sls/research/jupiter.shtml

Licencias de software libre compatibles con la GPL. (2014). Descargado Mayo, 2014, de http://www.gnu.org/licenses/license-list.html

Litman, D., y Silliman, S. (2004). ITSPOKE: an intelligent tutoring spoken dialogue system. En *Demonstration papers at HLT-NAACL* (Inf. Téc., pp. 5–8). Stroudsburg, PA, USA: Association for Computational Linguistics.

Llisterri, J. (2006). Introducción a los sistemas de diálogo. En J. Llisterri y M. J. Machuca (Eds.), *Los sistemas de diálogo* (Inf. Téc., pp. 11–21). Bellaterra - Soria: Universitat Autónoma de Barcelona -Fundación Duques de Soria. Descargado Mayo, 2014, de http://liceu.uab.cat/~joaquim/publicacions/Llisterri_06_Sistemas_Dialogo.pdf

López-Cózar, R. (2014). *Sistemas de diálogo hablado y multimodal.* Descargado Mayo, 2014, de http://www.ugr.es/~rlopezc/sistemas_dialogo.htm

Loquendo Text to Speech (TTS). (2014). Descargado Mayo, 2014, de http://www.nuance.com/for-business/by-solution/customer-service-solutions/solutions-services/inbound-solutions/loquendo-small-business-bundle/text-to-speech/index.htm

Mayani, P. (2012). *Android - Asynchronous image loading in Listview.* Descargado Mayo, 2014, de http://www.technotalkative.com/android-asynchronous-image-loading-in-listview/

Menezes, P., Lerasle, F., Dias, J., y Germa, T. (2007). Towards an interactive humanoid companion with visual tracking modalities. En M. Hackel (Ed.), *Humanoid robots: Human-like machines* (pp. 367–398). Vienna, Austria: Itech.

Mercury. (2013). Descargado Mayo, 2014, de http://groups.csail.mit.edu/sls/research/mercury.shtml

Metze, F., Wechsung, I., Schaffer, S., Seebode, J., y Moller, S. (2009). Reliable Evaluation of Multimodal Dialogue Systems. En J. Jacko (Ed.), *Human-Computer Interaction, Part II* (pp. 75–83). Berlin Heidelberg: Springer-Verlag.

Mostow, J. (2008). Experience from a reading tutor that listens: Evaluation purposes, excuses, and methods. En C. Kinzer y L. Verhoeven (Eds.), *Interactive Literacy Education: Facilitating Literacy Environments Through Technology* (pp. 117–148). New York: Erlbaum Publishers.

Natural vox. (2011). Descargado Mayo, 2014, de http://www.naturalvox.com/

Nimodia, C., y Deshmukh, H. (2012). Android Operating System. En *Software Engineering* (Vol. 3, pp. 10–13). doi: 2229-4007

Oracle. (2013a). Información general. En Oracle (Ed.), *MySQL 5.0 Reference Manual* (pp. 1–32). Descargado de http://dev.mysql.com/doc/refman/5.0/es/introduction.html

Oracle. (2013b). Sintaxis de sentencias SQL. En Oracle (Ed.), *MySQL 5.0 Reference Manual* (pp. 651–750). Descargado de http://dev.mysql.com/doc/refman/5.0/es/sql-syntax.html

Pavan, B. (2013). *Alternativas a Siri: asistentes personales para Android.* Descargado Mayo, 2014, de http://www.nuance.es/empresas/solucion/soluciones-de-atencion-al-cliente/servicios-y-soluciones/nuance-mobile-advantage/nina/index.htm

Pegasus. (2013). Descargado Mayo, 2014, de http://groups.csail.mit.edu/sls/research/pegasus.shtml

Pérez, A. (2011). Las Acciones de Voz para Android en español. En *Blog Oficial de Google España.* Descargado Mayo, 2014, de http://googleespana.blogspot.com.es/2011/09/las-acciones-de-voz-para-android-en.html

Pérez, G., Amores, G., y Manchón, P. (2006, diciembre). A multimodal architecture for home control by disabled users. En *Spoken Language Technology Workshop* (pp. 134–137). Palm Beach, Aruba. doi: 10.1109/SLT.2006.326836

Plan comparison. (2014). Descargado Mayo, 2014, de https://x10hosting.com/wiki/Plan_comparison

Raphael, J. (2012). 70 things to try with Google's Android 4.1 Voice Search. Descargado Mayo, 2014, de http://www.itworld.com/personal-tech/288505/70-things-try-googles-android-41-voice-search?page=0,0

Rodrigo, G. (2012). Mejores Asistentes de Voz para Android. Descargado Mayo, 2014, de http://androidzone.org/2012/12/mejores-asistentes-de-voz-para-android

Shanklin, W. (2011). Google is close to releasing Siri rival called Majel. Descargado Mayo, 2014, de http://www.geek.com/apple/google-is-close-to-releasing-siri-rival-called-majel-1449289/

Siri. (2014). Descargado Mayo, 2014, de http://www.apple.com/es/ios/siri/

Soto, J. (2014). Reconocer la voz. Descargado Mayo, 2014, de http://www.negomobile.es/en/node/25

SQlite. (2014). Descargado Mayo, 2014, de http://www.sqlite.org/

Texas Instruments. (2012). Text-To-Speech-and-Speech-Recognition-on-Android. Descargado Mayo, 2014, de http://processors.wiki.ti.com/index.php/Text-To-Speech-and-Speech-Recognition-on-Android

Trivi, J. (2009). An introduction to Text-To-Speech in android. En *Android Developers Blog.* Descargado Mayo, 2014, de http://android-developers.blogspot.com.es/2009/09/introduction-to-text-to-speech-in.html

Turunen, M., Salonen, E.-P., Hartikainen, M., Hakulinen, J., Jokinen, K., Rissanen, J., y Antti, K. (2004). *AthosMail: a Multilingual Adaptive Spoken Dialogue System for the E-mail Domain.* Proceedings of the COLING Workshop Robust and Adaptive Information Processing for Mobile Speech Interfaces, Geneva, Switzerland.

Universidad Carlos III de Madrid. (2014). *Plantilla presupuesto del proyecto fin de carrera.* Descargado Mayo, 2014, de http://www.uc3m.es/portal/page/portal/administracion_campus_leganes_est_cg/proyecto_fin_carrera

Vaquero, C., Saz, O., Lleida, E., Marcos, J., y Canalís, C. (2006). VOCALIZA: An application for computer-aided speech therapy in spanish language. En *IV Jornadas en Tecnología del Habla* (pp. 321–326). Zaragoza, España.

Verbio Speech Technologies. (2014). Descargado Mayo, 2014, de http://www.verbio.com/webverbio3/

Vico, A. (2011). Arquitectura de Android. *La Columna 80*. Descargado Mayo, 2014, de http://columna80.wordpress.com/2011/02/17/arquitectura-de-android/

Vogel, L. (2010). *Android sqlite database and content provider - tutorial.* Descargado Mayo, 2014, de http://www.vogella.com/articles/AndroidSQLite/article.html#overview_sqliteandroid

Voice IME. (2014). Descargado Mayo, 2014, de https://google-voice-typing-integration.googlecode.com/git/VoiceImeUtils/doc/index.html?com/google/android/voiceime

Voice Ticketing. (2011). Descargado Mayo, 2014, de http://www.voiceweb.eu/services/speech_vas/ticketing/products2/voice_ticketing

x10hosting. (2014). Descargado Mayo, 2014, de https://www.x10hosting.com/

Zanolin, L. (2011). Add Voice Typing To Your IME. En *Android Developers Blog.* Descargado Mayo, 2014, de http://android-developers.blogspot.com.es/2011/12/add-voice-typing-to-your-ime.html